NoAW project

Innovative approaches to turn agricultural waste into ecological and economic assets

Deliverable n°: 2.4

Deliverable title: Argumentation software programming and data analysis

Planned delivery date (as in DoA): 31/03/2019
Actual submission date: 17/06/2019 (M33)

Workpackage: WP2
Workpackage leader: Anna Ekman Nilsson (RISE)
Deliverable leader: INRA
Dissemination level: PU
EC Version: V1

Research and Innovation action: GA no. 688338

Start date of the project: October 1st, 2016

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

Table of Contents

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

1. Document Information

Author(s)

Organisation name lead contractor	INRA

Author	Organisation	e-mail
Abdelraouf Hecham	LIRMM	hecham.abdelraouf@gmail.com
Martin Jedwabny	INRA	martinjedwabny@gmail.com
Pierre Bisquert	INRA	pierre.bisquert@inra.fr
Patrice Buche	INRA	patrice.buche@inra.fr
Madalina Croitoru	UM	madalina.croitoru@lirmm.fr

Revision history

Version	Date	Modified by	Comments
1	07/05/2019	Abdelraouf Hecham	Starting version
2	07/06/2019	Patrice Buche et Pierre Bisquert	Final version

Dissemination level

This deliverable is part of a project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 688338	
Dissemination Level	
PU Public	**PU**
CI Classified, as referred to Commission Decision 2001/844/EC	
CO Confidential, only for members of the consortium (including the Commission Services)	

2. Summary

Background	The **NoAW** project's goal is driven by a "near zero-waste" society requirement and focuses in the development of innovative efficient approaches that allow the conversion of growing agricultural waste issues into eco-efficient bio-based products. These approaches aim for direct benefits for the environment, the economy and the EU consumer. Nonetheless, one major challenge is the selection of these new waste valorization routes. This is generally done through Life Cycle Analysis (LCA) methods where each valorization route is assessed through a set of impact categories (e.g. the impact of a valorization route on global warming, water consumption, land use, etc.). **Where does this deliverable fall within the NoAW project?** This deliverable 2.4 falls within the work package WP2, which is focused on the multi-criteria evaluation of different waste valorization routes. The aim of WP2 is to use LCA methods to establish the most interesting valorization routes from an environmental point of view. This analysis requires a ranking of importance of the impact categories which is normally provided by default by the chosen LCA method. The proposal of this deliverable 2.4 comes before the LCA methods to assess how the different stakeholders value the impact categories using collective decision-making techniques.
Objectives	The objective of this deliverable is to propose and implement a workflow for collective decision-making i.e. the fair aggregation of individual preferences of stakeholders based on their justifications. **What is the aim of this deliverable within the NoAW project?** More precisely, we are interested in establishing a ranking of the different impact categories based on the stakeholders' opinions (preferences). This ranking would then be used for the LCA methods within WP2. In order to achieve that, we refer to Computational Social Choice and Argumentation techniques and methods. **Why would the aggregation of justified preferences be useful for NoAW?** Aggregating the stakeholders' preferences while taking into account their justification allows for a contextualized and fair collective assessment of the importance of the impact categories. If this is applied to LCA methods, it might yield a more relevant ranking of the valorization routes for the NoAW project.
Methods	We propose an approach that combines preference aggregation methods such as voting rules with justification analysis methods based on argumentation. The workflow of the proposal is to first elicit the stakeholders (a.k.a. agents) preferences and their justifications (i.e. the reasoning steps that lead to a preference), then to automatically detect agreements and disagreements between the reasoning steps provided by these agents. Afterwards, they can discuss the points of conflicts and potentially change their rankings or justifications. Next, the argumentation technique known as Statement Graphs is applied to establish accepted,

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

	ambiguous, and rejected justifications which would be later used to filter and update the initial preferences using a proposed "rank sliding" approach. Then a voting method from social choice can be used to aggregate these updated ranks.
Results & implications	In this deliverable we propose an approach and a set of software tools for collective decision-making by evaluating the justifications of the rankings on different alternatives. The main results of this deliverable are the following: ➢ Theoretical proposals on justification analysis and how justifications can be taken into account in preference aggregation. Namely, a detailed workflow for the elicitation of preferences and their justifications, automatic reasoning about these justifications and assessment of their validity, and updating and aggregating the rankings accordingly. ➢ DAMN (Defeasible Reasoning With Statement Graph) tool, which is a web-based tool that provides a visual description of the points of agreement and disagreement between participants along with a user collaboration feature. ➢ ELDR (Existential Logic for Defeasible Reasoning) tool, which is a logic-based tool for reasoning with contradictory knowledge. ➢ Rank Sliding tool which allows the change of preference based on the accepted and rejected justifications. ➢ Compared to the tools of the state of the art, DAMN/ELDR is the only one which support simultaneously a set of defeasible reasoning features, multi-agent collaboration and visualization. **What are the results and implications for the NoAW project?** While the proposed approach and tools were designed to work in the general case, we apply them on the assessment of the importance of impact categories in LCA within NoAW. More precisely, a survey on 31 participants from the European projects NoAW and Agrocycle was conducted on LCA impact categories where the stakeholders expressed their preferences and justifications. We show that the proposed method is relevant in producing a justified ranking of the impact categories. The NoAW specific results are: ➢ The input data and output files containing the results for NoAW impact indicators (survey data, logical representation of the justifications, original rankings, updated rankings, final aggregation). These documents are available at https://data.inra.fr/dataset.xhtml?persistentId=doi:10.15454/XVL4BA ➢ The ontology (i.e. a formal structured representation) representing the reasons justifying the rankings associated with LCA indicators.

Abbreviations used in the report

Abbreviation	Full name
API	Application Programming Interface
DAMN	Defeasible Reasoning With Statement Graph
DLGP	DataLoG +
ELDR	Existential Logic for Defeasible Reasoning
FAIR	Findable, Accessible, Interoperable, Reusa-ble
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
LCA	Life Cycle Assessment
noSQL	Not Only SQL
REST	representational state transfer
SG	Statement Graph
SQL	Standard Query Language

This project has received funding from the European Union's Horizon 2020 research and innovation pro-gramme under grant agreement No 688338

3. Introduction

Collective decision-making and preference aggregation are widely used in today's society, from formal ways as in political elections to informal ones as choosing a restaurant between a group of friends. In all these cases, the individuals (also called agents) express their preferences over a set of options called alternatives, and the objective is to fairly aggregate these preferences (also called rankings) into a collective ranking and thus obtain a decision which satisfies the group as a whole. This can be done using classical social choice theory where the aggregation of the preferences is computed by a voting rule. The deliverable D1.3 of the NoAW project proposed a tool that is able to aggregate the preferences of agents using various voting methods and offers the ability to compute this aggregation given a grouping criterion such as by country, by occupation, etc.

Collective decision making is mostly useful when the preferences differ from an agent to another, as it becomes trivial if all agents agree on the same preferences. A fair aggregation method is then required in order to solve these differences of preferences. While aggregating preferences provides a direct answer to the collective decision-making problem, it only offers a relatively superficial solution with various short comings such as the inability to explain why agents' preferences differ in the first place. If we seek better and more flexible collective decision-making, then we need to delve deeper and consider the justifications for each agent's preferences.

A justification is a set of arguments that an agent puts forward to describe the reasoning behind his ranking of alternatives. These arguments can be seen as a set of "reasoning steps" to reach a conclusion. Agents may agree or contradict one another on one or many of these reasoning steps. We believe that taking into account these justifications when aggregating the preferences would yield a better final result. Indeed, this limits the impact of personal subjective preferences in favor of more grounded ones, ensuring a more rational decision. In this case, the problem we are considering would combine both Computational Social Choice (for preference aggregation using voting methods) and Argumentation (for justifications analysis). Note that there has been significant research towards decision-making on both of these fields independently. Social choice theory has been integrated in the analysis of some popular aggregation methods in multi-criteria decision aiding, i.e., the ordinal methods are based on the Condorcet method, e.g., [Roy, 1991], and the cardinal ones are based on the Borda method, e.g., [Von Winterfeldt et al., 1986]. On the other hand, several proposals towards the usage of argumentation in decision-making have been proposed, notably the one by [Amgoud et al., 2009] which proposes an abstract argumentation-based framework with a 2-step procedure where at first the arguments for beliefs and options, and the conflicts between them, are built and at the second step we have pairwise comparisons of the options using decision principles.

Therefore, the problem we are considering is to obtain justified collective decision-making on alternatives through automatic justification analysis. This can be achieved by combining computational social choice and argumentation techniques. The idea is to first ask the agents to provide a justification for each of their preferences, then to automatically detect agreements and contradictions between the reasoning steps provided by these agents. Afterwards, agents can discuss the points of disagreements and potentially change their rankings or justifications. Next, argumentation techniques can be applied to establish accepted and rejected justifications which would be used to filter and update the initial preferences. Then a voting method from social choice can be used to aggregate these updated preferences.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

The analysis of the justifications for preferences offers various advantages: First the ability to base decisions on justified knowledge rather than personal preferences, this might allow agents to identify hidden biases by pushing them to justify their preferences. Second, justifications can be used as an explanation for the final decision by showing the points of agreements and disagreements between the different agents. Third, the automatic reasoning on justification would showcase the logical consequences that can be derived from an agent's knowledge, this might help prevent substantive irrationality [Bisquert et al., 2016] where an agent does not see the contradictions in his own knowledge. Finally, by explicitly representing the justifications, the final decision becomes flexible towards new knowledge as the automatic reasoning will showcase the changes in accepted and rejected justifications given new information. In particular, if new information becomes available after a decision has been made, we can automatically check how it would have affected the final result.

The first purpose of this report is to describe our proposed approach and its supporting tools. We describe the problem, the overall objective, and the theoretical aspect of our proposal, then we detail the implementation of the different tools. Our second purpose is to demonstrate our approach by applying it to the data collected within the NoAW project where different agents gave their preferences on a set of LCA (Life Cycle Analysis) impact categories (e.g. water consumption, land use, global warming, etc.) along with their justifications. We present the aggregated preference with and without taking into account the justification analysis. Note that this use-case is not restrictive on the applicative usage of the proposed platform and set of tools as we always consider a more general setting for other future possible applications of our approach.

In a nutshell, our overall objective is to obtain **justified collective decision-making on alternatives through justification analysis** which would allow us to:

1. Base decisions on justified knowledge rather than personal preferences.

2. Provide explanation for these decisions.

3. Showcase the points of disagreements between the different agents

4. Showcase the logical contradictions in an agent's knowledge.

5. Provide flexibility toward new knowledge.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

4. Proposal and Results

4.1. Problem Description and Overall Objective

The general context is that of collective decision-making over a set of options called *alternatives* with aim of obtaining a *justified collective ranking* of these alternatives from best to worst or from most to least important. These alternatives can represent for example the impact categories of life cycle analysis (e.g. land use, water consumption, etc.).

This deliverable D2.4 along with the deliverable D1.3 fall within the overall objective of aggregating preferences of different agents over a set of alternatives while taking into account the justifications they provide for these preferences. More formally, the problem is expressed as follows:

- **Input:** A set of options *A* called *alternatives* and a set of *agents N*. Each agent has a *knowledge base* where he expresses his *preferences* (a ranking on the set of alternatives) along with a *justification* (a set of facts and rules that justify these preferences).

- **Output:** A justified collective ranking of alternatives obtained by the aggregation of all the preferences of the agents while taking into account their justifications.

Example 1. *Within the NoAW project, suppose we have a set of impact categories that represent the considered alternatives A = {land use, water consumption, global warming}. We want to obtain an aggregated ranking of these categories (from most important to least important) based on the rankings of the agents N = {Alice, Bob}. The knowledge of these agents is as follows:*

- *Alice's ranking is* ***waterConsumption > globalWarming > landUse*** (> means "is preferred to"). *This agent argues that water consumption is ranked 1 because it is a global issue especially with the limited supply of fresh or usable water. Global warming is ranked 2 because it is hard to control, and land use is ranked 3 because it has a limited weak impact on food supply compared to the other two alternatives.*
- *Bob's ranking is* **globalWarming > landUse > waterConsumption***. This agent argues that global warming is ranked 1 because it is almost an irreversible outcome and should be the highest priority as an impact category. Land use is ranked 2 because it has a strong impact on food supply, and water consumption is ranked 3 because it is a local problem and a solution is available through recycling.*

The output can be for example "Global warming > Land Use > Water consumption" based on some preference aggregation (e.g. voting rule) and a justification handling technique.

This overall objective can be split into two main problems (cf. Figure 1): **preference aggregation** and **justification analysis**. In the **deliverable D1.3** titled *"Computational Social Choice software programming and data analysis"* a solution to the preference aggregation problem has been proposed. It relies on the classical social choice theory where the aggregation of the preferences is computed by a *voting rule.* The input in this case is the set of alternatives and the agents' rankings. The output is a collective aggregated ranking of the alternatives.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

In this **deliverable D2.4** titled *"Argumentation software programming and data analysis"* we tackle the problem of justification analysis which comes with its own set of sub-problems: automatic detection of conflicts in the justifications of the agents, establishing accepted justifications by solving these conflicts, and reflecting these accepted justifications by updating the agents' rankings to produce justified preferences. The input in this case is the set of alternatives and the knowledge base of each agent containing his or her ranking along with its justifications. The output is an updated justified ranking for each agent.

Hence, the decision problem of this deliverable can be summarized by the following questions:

1. How to automatically reason and detect conflicts in justifications?

2. How to solve these conflicts and establish the accepted justifications?

3. How to update the rankings of each agent to produce justified rankings?

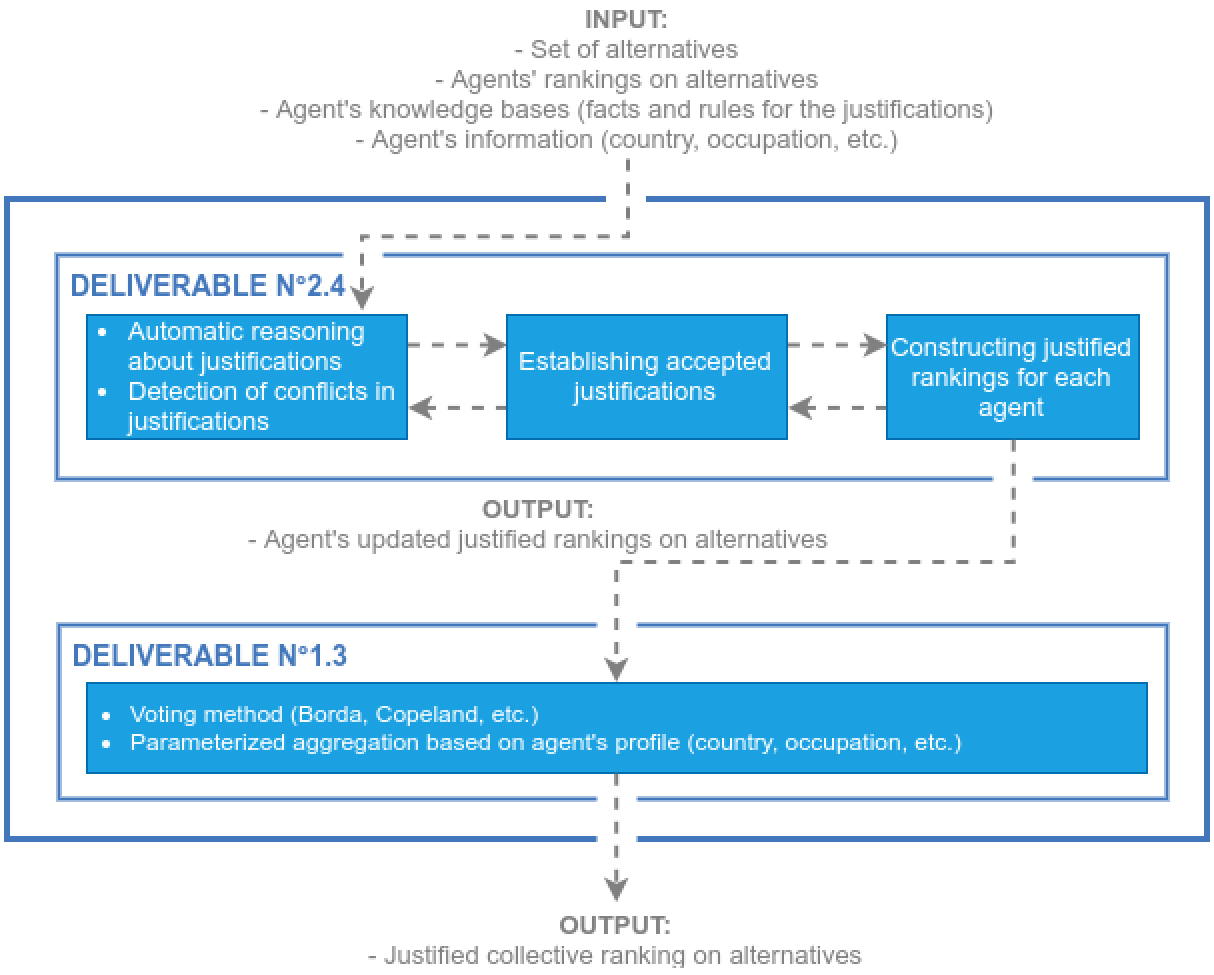

INPUT:
- Set of alternatives
- Agents' rankings on alternatives
- Agent's knowledge bases (facts and rules for the justifications)
- Agent's information (country, occupation, etc.)
DELIVERABLE N°2.4
Automatic reasoning about justifications
Detection of conflicts in justifications
Establishing accepted justifications
Constructing justified rankings for each agent
OUTPUT:
- Agent's updated justified rankings on alternatives
DELIVERABLE N°1.3
Voting method (Borda, Copeland, etc.)
Parameterized aggregation based on agent's profile (country, occupation, etc.)
OUTPUT:
- Justified collective ranking on alternatives

Figure 1. Overall architecture for justified collective decision-making on alternatives

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

4.2. General approach for justification analysis

In this section, we describe the theoretical approach to tackle the questions presented in the previous section.

4.2.1. Tackling the automatic reasoning about justifications

A justification is a set of arguments that an agent puts forward to describe the reasoning behind his/her ranking of alternatives. These arguments can be seen as a set of “reasoning steps” to reach a conclusion. Agents may agree or contradict one another on one or many of these reasoning steps. In order to automatically build and detect conflicts between the different justifications, we rely on a logical language and inference mechanism.

Logic is the systematic study of valid inference. Typically, it uses facts to represent knowledge about the world, and rules to express deductive links between this knowledge. Applying rules on facts generates reasoning steps in the form of “given a **hypothesis** we deduce a **conclusion**”. For instance, suppose we have a factual knowledge stating that “kowalski” is a penguin, we also know the rule that penguins are birds, therefore we can generate the reasoning step that states: *“given that kowalski is a penguin, we deduce that he is a bird”*.

Thus in order to apply the inference mechanisms of logic on justifications we need to express these justifications in a logical language representing facts and rules (as shown in the following Example 2). Different logical languages have been defined [Baader et al., 2005, Cali et al., 2010], we chose to use the existential rules logical language [Cali et al., 2010] which is a first order logical fragment used in the Semantic Web and Databases domain. This would allow us to supplement the justifications expressed by the agents with knowledge extracted from the web or from domain-specific databases (such as databases containing the results of agricultural experiments or survey data).

In this logical language, knowledge is expressed using **atoms** which describe a property of some individuals. For example, “kowalski is a penguin” is expressed as the atom “penguin(kowalski)”. Atoms can be linked together via the **conjunction** “∧” and the **implication** “⇒” connectors. Conjunctions express a combined knowledge, for example, “kowalski is a 2 years old penguin” is represented as “penguin(kowalski)∧age(kowalski,2)”. Implication express a rule, for example, “all penguins are birds who do not fly” is represented as “∀ X penguin(X) ⇒bird(X) ∧ notFly(X)” (which translates to “for all individuals X, if X is a penguin then X is a bird and it does not fly”). The logical language allows also the representation of **tautology** “⊤” and **falsity** “⊥”. Tautology indicates that some knowledge is unconditionally true, it is used to represent **factual knowledge** (facts), for instance, the fact “kowalski is a penguin” is represented as “⊤ ⇒ penguin(kowalski)”. Falsity indicates contradictions, for example, the knowledge that “an animal cannot fly and not fly at the same time” is represented as “∀ X fly(X) ∧ notFly(X) ⇒⊥” (which translates to “for all individuals X, if X flies and X does not fly, then we can deduce a contradiction”).

Example 2. *Consider the following simple example where two agents want to know if “kowalski” flies (i.e. the query is “fly(kowalski)”). Each agent has his own knowledge base and they share a common knowledge.*

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 688338

- *Common knowledge base: {⊤ ⇒penguin(kowalski), ∀ X fly(X) ∧ notFly(X) ⇒⊥} (kowalski is a penguin, and one cannot fly and not fly at the same time).*
- *Alice's knowledge base: { r_1: ∀ X penguin(X) ⇒bird(X) ∧ notFly(X) } (penguins are birds who do not fly. This rule is given the label r_1 so it can be easily referenced).*
- *Bob's knowledge base: { r_2: ∀ X bird(X) ⇒fly(X) } (birds fly).*

From this combined knowledge we can generate the following reasoning steps (rule applications):

- *By applying r_1 we generate: "penguin(kowlaski) ⇒bird(kowalski) ∧notFly(kowalski)"*
- *By applying r_2 we generate: "bird(kowalski) ⇒fly(kowalski)"*

Representing the justifications in a logical language allows us to identify the links between the different reasoning steps (**support links** when the conclusion of a rule application is used in the hypothesis of another rule application, and **attack links** when the conclusion of rule application contradicts the hypothesis of another rule application). These links are represented using Statement Graphs [Hecham et al., 2018a] where each node (also called statement) represents a reasoning step (i.e. a rule application), dashed arrows represent support links while normal arrows represent attack links. The statement "*→ ⊤(true)*" which has an "accepted strictly" status (i.e. cannot be rejected) represents the concept of truth and is used to supports factual knowledge. For instance, the previous example can be represented in the Statement Graph depicted in Figure 2.

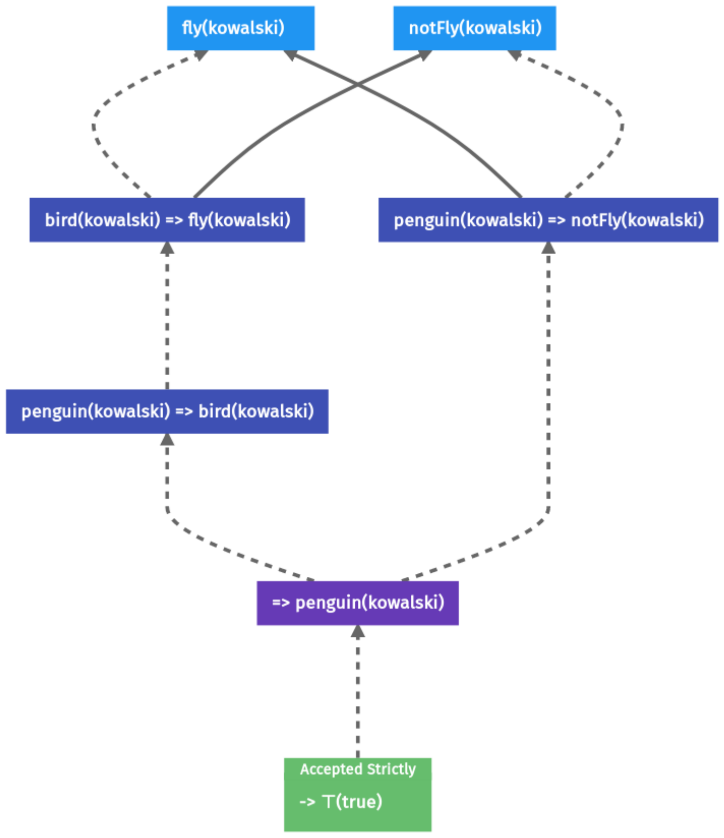

fly(kowalski)
notFly(kowalski)
bird(kowalski) => fly(kowalski)
penguin(kowalski) => notFly(kowalski)
penguin(kowalski) => bird(kowalski)
=> penguin(kowalski)
Accepted Strictly
-> ⊤(true)

Figure 2. Statement Graph of Example 2.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

By using the Statement Graph formalism, we can automatically generate the reasoning steps and build the support and attack links representing agreements and contradictions which will be used afterward to establish if a justification is accepted or rejected.

4.2.2. Establishing accepted and rejected justifications

Accepting or rejecting a reasoning step is achieved using a logical semantics. Classical logic semantics fail in presence of contradictions due to the principle of explosion *"from falsehood, anything follows"* [Carnielli et al., 2001]. In fact, if classical logic semantics is applied to contradicting justifications it will yield that all justifications are accepted and reasoning becomes trivial. That is why we need to use a conflict-tolerant semantics.

Defeasible reasoning using Statement Graphs is an efficient form of conflict-tolerant reasoning [Hecham et al., 2018a]. It allows the expression of agents' knowledge as a set of facts and rules along with a superiority relation between rules. Superiority relation (denoted with ">>") expresses that certain rules override other rules, for example: "the rule stating that penguins are non-flying birds overrides the rule stating that if it is a bird then it flies". This superiority relation also allows us to define outcomes scenario where we can check what justifications are accepted if we consider the knowledge of one type of agents as superior to other agents.

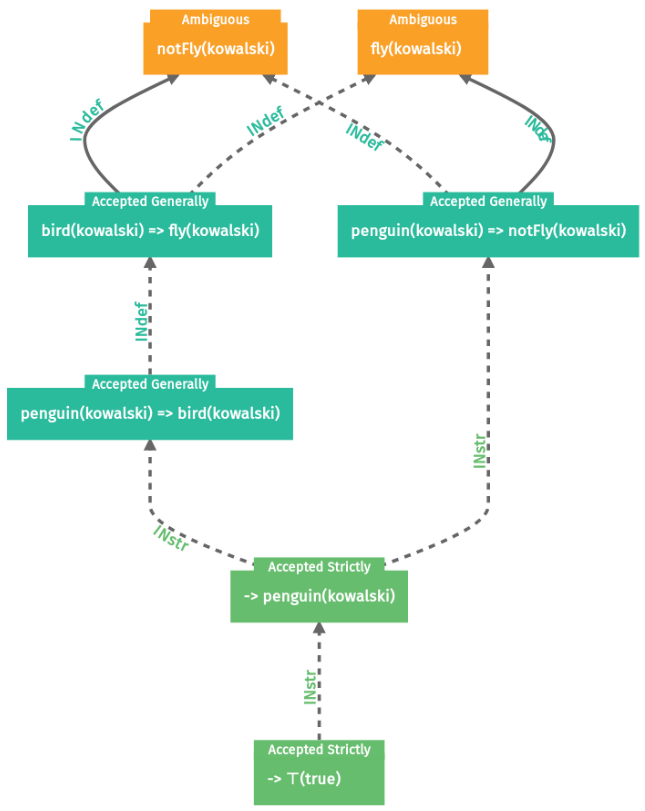
Defeasible reasoning comes with different semantics, the interested reader is referred to [Antornio et al., 2000] for a detailed description. For simplicity, we will only describe a simplified version of defeasible reasoning called *"ambiguity propagation without team defeat"* [Hecham et al., 2018a]. In its simplest form, this semantics affects a label to each node of the Statement Graph, this label can either be IN (accepted), OUT (rejected), or AMBIG (ambiguous).

The intuition behind the IN, OUT, AMBIG labels is as follows:

- A statement is labeled **IN** (accepted) if all atoms in its hypothesis are supported by accepted statements and none of them is attacked by an accepted statement.
- A statement is labeled **OUT** (rejected) if one of the atoms in its hypothesis is not supported by an accepted statement or it is attacked by an accepted statement with a superior rule.
- A statement is labeled **AMBIG** (ambiguous) if one of the atoms in its hypothesis is supported by an accepted statements and it is attacked by an IN statement or it is only supported by and ambiguous statement.

Example 3. *Consider the decision problem of Example 2. Figure 3.1 showcases the result of applying the semantics on the Statement Graph of Example 2. We can observe that the status of "fly(kowalski)" and "notFly(kowalski)" is* ***ambiguous*** *because they are both supported and attacked by accepted statements. Now suppose the data engineer agrees that "the rule stating that penguins are non-flying birds overrides the rule stating that if it is a bird then it flies", in this case he/she favors the knowledge expressed by Alice over the one expressed by Bob by adding the superiority relation "$r_1 >> r_2$" to the common knowledge base. The resulting Statement Graph is described in Figure 3.2. We can observe in this case that "fly(kowalski)" is now rejected because it is attacked by an accepted superior rule application, and "notFly(kowalski)" is now accepted.*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338



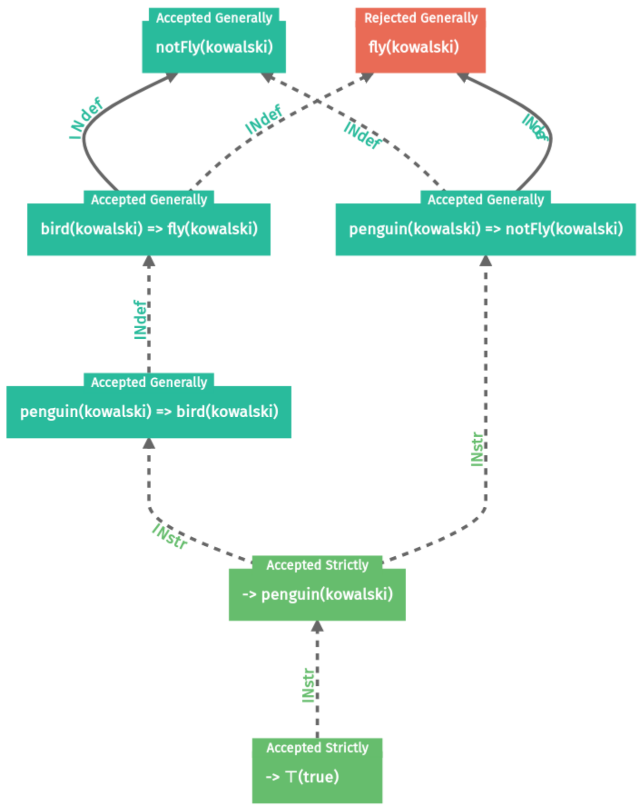


Figure 3.1. SG of Ex.2 without superiority relation. **Figure 3.2** SG of Ex.2 with superiority relation.

By using the defeasible reasoning semantics, we can obtain the status of the different reasoning steps. This would later need to be reflected in the agents' rankings.

4.2.3. Updating the rankings to produced justified rankings

Our general aim is to produce justified rankings. Once the status of justifications has been established, we need to update each agent's rankings to reflect only *"admissible"* rankings. By "admissible" rankings we mean a ranking that either has an accepted justification or a reasonably contested (i.e. "ambiguous") justification. Therefore, we need to properly define admissible rankings by answering the following questions: what is a reasonably contested justification? And how should rankings that are unjustified or have rejected/ambiguous justification be handled? We propose two approaches to answer these questions, the "**collaboration approach**" and the **"rank sliding approach"**.

Collaboration approach. The idea is to have the agents discuss their points of views and change their justifications or their rankings based on the contradictions that our automatic reasoning platform identifies. This discussion phase might push agents to change their rankings or justifications and result in only admissible rankings with accepted justifications. In this case, these justified rankings are transmitted to the voting module (deliverable D1.3) and a final decision is reached as illustrated in the following example.

Example 4. Consider part of the data extracted for NoAW project which is geared towards agro-waste valorization. The agents (Alice and Bob) want to decide between using the agro-waste from producing wine for either "Fertilization" or send it to "Distillery".

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

- *Alice argues that “Fertilization” should be ranked 1 because it reduces artificial fertilizers while “Distillery” is ranked 2 because it does not reduce artificial fertilizers.*
- *Bob argues that “Distillery” should be ranked 1 because sending wine by-products to distilleries is mandatory in France, and “Fertilization” is ranked 2 because it is not mandatory.*
- *However, Alice argues back that sending wine by-products to distilleries is no longer mandatory.*

The logical representation of this knowledge is as follows:

- *Common knowledge base: {∀ X mandatory(X) ∧ notMandatory(X) ⇒ ⊥ } (something cannot be mandatory and not mandatory at the same time).*
- *Alice’s knowledge base: { r_1: ⊤ ⇒ reducesArtificialFertilizers(fertilization) ∧ notReducesArtificialFertilizers(distillery), r_2: ∀ X reducesArtificialFertilizers(X) ⇒rank(X,1), r_3: ∀ X notReducesArtificialFertilizers(X) ⇒rank(X, 2), r_4: ⊤ ⇒notMandatory(distillery)}*
- *Bob’s knowledge base: { r_1: ⊤ ⇒mandatory(distillery) ∧ notMandatory(fertilization),*

  r2: ∀ X mandatory(X) ⇒rank(X,1), r3: ∀ X notMandatory(X) ⇒rank(X, 2)}

The resulting Statement Graph about the ranking of distillery along with the statuses of the justifications is shown in Figure 4

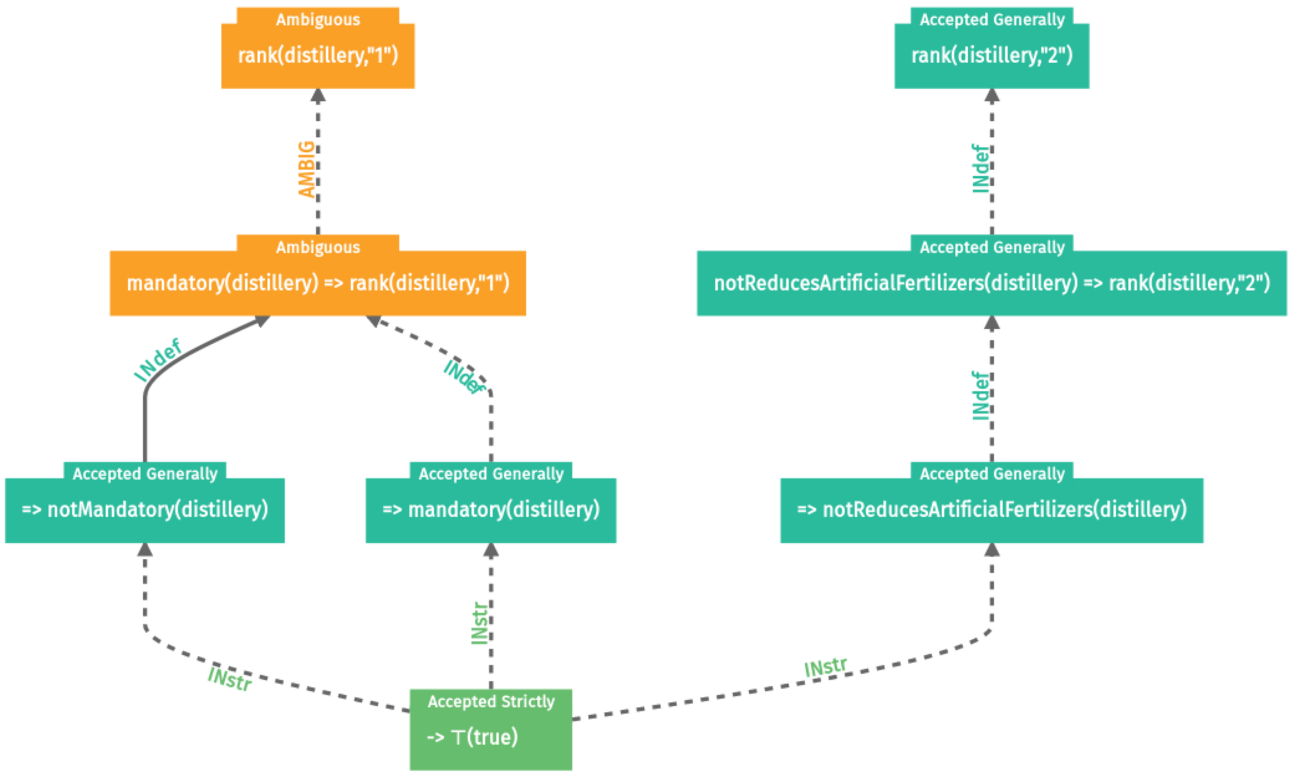


Figure 4. *Statement Graph for ranks on “Distillery” of Example 4.*

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 688338

Without considering justifications, the ranking of Alice would be "Fertilization > Distillery" and the ranking of Bob would be "Distillery > Fertilization". If we aggregate these rankings through a voting method, we obtain that both options are equivalent ***"Fertilization ~ Distillery"****.*

However, suppose that by engaging in the discussion phase Bob is now convinced of Alice's argument that sending wine by-product to distilleries is no longer mandatory and changes his rankings to "Fertilization > Distillery". The resulting aggregated rankings is ***"Fertilization > Distillery"****.*

The collaboration approach is a favorable approach because it allows agents to discuss their justifications, and conflicts might be resolved before a vote even happens. However, if the agents are not available to engage in a discussion phase, or even after discussing they still end up with ambiguously justified rankings, then we need another approach that handles such situations.

Rank sliding approach. The idea is to change the rankings of agents (if the status of their justification is rejected or ambiguous) by "sliding" their ranks to the closest rank with accepted justification expressed by any one of the agents. It means that a vote with rejected or ambiguous justification is considered similar toits closest justified vote. To avoid arbitrary changes of rankings we defined a set of desirable properties for a rank sliding approach:

- The approach should change inadmissible ranks towards the "best" admissible rank.
- The approach should lead to the least amount of rank changes.
- The approach should be as precautious as possible and favor a change towards higher ranks.
- The approach should give an incentive to agents to not only justify their rankings but also to either support or contradict other agents' justifications.

Within these guidelines, we propose a rank sliding approach called **skeptical rank sliding** approach with the following conditions:

1. A rank is admissible if it has an accepted justification or if it has an ambiguous justification and there is no rank with an accepted justification.
2. A rank with an ambiguous justification is changed to the closest rank with an accepted justification if it exists. If two ranks with accepted justification are at the same distance from this rank, then it is changed to the closest higher rank with accepted justification. Otherwise, if there is no rank with an accepted justification then this rank remains unchanged.
3. A rank with a rejected justification or without a justification is simply dismissed.

Example 5. *Consider a use case in the NoAW project, suppose we have a set of impact categories that represent the considered alternatives A = { landUse, waterConsumption, globalWarming }. We want to obtain an aggregated ranking of these categories (from most important to least important) based on the rankings of the agents N = { Alice, Bob, Carol }. The knowledge of these agents is as follows:*

- Alice's ranking is **waterConsumption > globalWarming > landUse**. This agent argues that waterConsumption is ranked 1 because it is a global issue especially with the limited supply of fresh or usable water. GlobalWarming is ranked 2 because it is hard to control, and landUse is

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

ranked 3 because it has a limited weak impact on food supply compared to the other two alternatives.

- Bob's ranking is **globalWarming > landUse > waterConsumption**. This agent argues that globalWarming is ranked 1 because it is almost an irreversible outcome and should be the highest priority as an impact category. LandUse is ranked 2 because it has a strong impact on food supply. WaterConsumption is ranked 3 because it is a local problem and a solution is available through recycling.
- Carol ranking is **landUse > waterConsumption > globalWarming**. She does not justify her ranking.

The logical representation of this knowledge is as follows:

- *Common knowledge base: {∀ X,Y impact(X,Y,local) ∧ impact(X,Y,global) ⇒ ⊥, ∀ X,Y impact(X,Y,strong) ∧ impact(X,Y,weak) ⇒ ⊥, ∀ X suppress(unjustified), unjustified(X) ⇒ ⊥, ⊤ ⇒ suppress(unjustified) } (the "suppress(unjustified)" atom is used to reject any unjustified rank)*
- *Alice's knowledge base: {⊤ ⇒ impact(waterConsumption,water,global) ∧ status(globalWarming,hardToControl) ∧ impact(landUse,foodSupply,weak),*

  impact(waterConsumption,water,global) ⇒ rank(waterConsumption,1), status(globalWarming,hardToControl) ⇒ rank(globalWarming, 2), impact(landUse,foodSupply,weak) ⇒ rank(landUse,3) }
- *Bob's knowledge base: {r$_1$: ⊤ ⇒ status(globalWarming,almostIrreversible) ∧ impact(landUse,foodSupply,strong) ∧ impact(waterConsumption,water,local) ∧solution(waterConsumption,available), status(globalWarming,almostIrreversible) ⇒ rank(globalWarming, 1), impact(landUse,foodSupply,strong) ⇒rank(landUse, 2) solution(waterConsumption,available) ⇒ rank(waterConsumption, 3)}*
- *Carol's knowledge base: { ⊤ ⇒unjustified(landUse) ∧ unjustified(waterConsumption) ∧ unjustified(globalWarming), unjustified(landUse) ⇒rank(landUse,1), unjustified(waterConsumption) ⇒ rank(waterConsumption,2), unjustified(globalWarming) ⇒rank(globalWarming,3)}*
- *The resulting Statement Graphs along with the statuses of the justifications for each rank of alternative is shown in Figure 5,6, and 7.*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

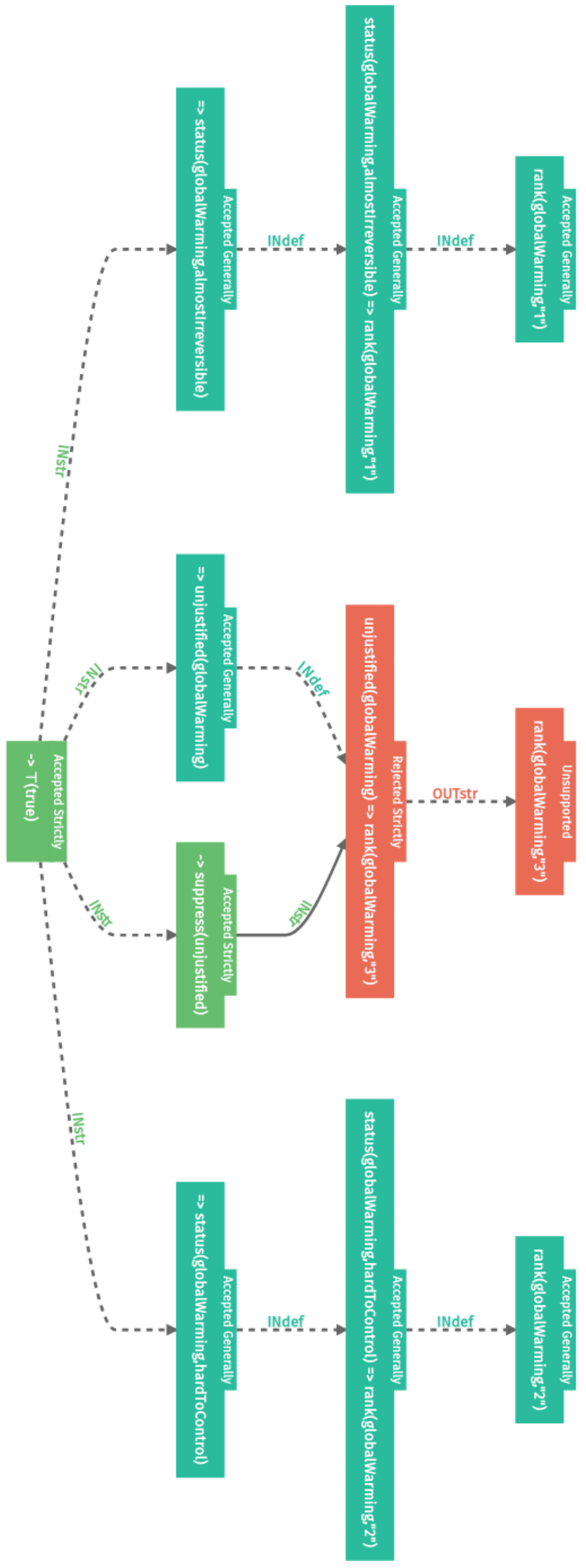


Figure 5. *Statement Graph for the globalWarming ranking and justifications.*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

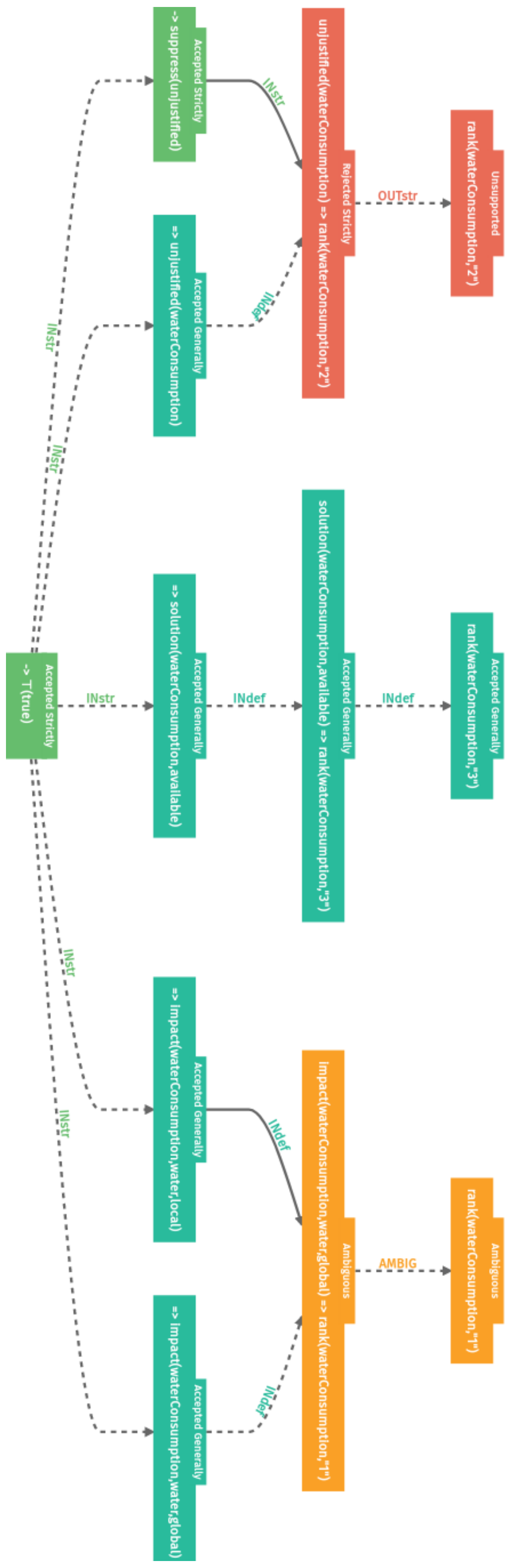

Accepted strictly
=> T(true)
Accepted strictly
=> supress(unjustified)
Accepted generally
=> unjustified(waterConsumption)
Rejected strictly
unjustified(waterConsumption) => rank(waterConsumption,"2")
Unsupported
rank(waterConsumption,"2")
Accepted generally
=> solution(waterConsumption,available)
Accepted generally
solution(waterConsumption,available) => rank(waterConsumption,"3")
Accepted generally
rank(waterConsumption,"3")
Accepted generally
=> impact(waterConsumption,water,local)
Accepted generally
=> impact(waterConsumption,water,global)
Ambiguous
impact(waterConsumption,water,global) => rank(waterConsumption,"1")
Ambiguous
rank(waterConsumption,"1")
INstr
INstr
INstr
INstr
INstr
INstr
INdef
INdef
INdef
INdef
INdef
OUTstr
AMBIG

Figure 6. *Statement Graph for the waterConsumption ranking and justifications.*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

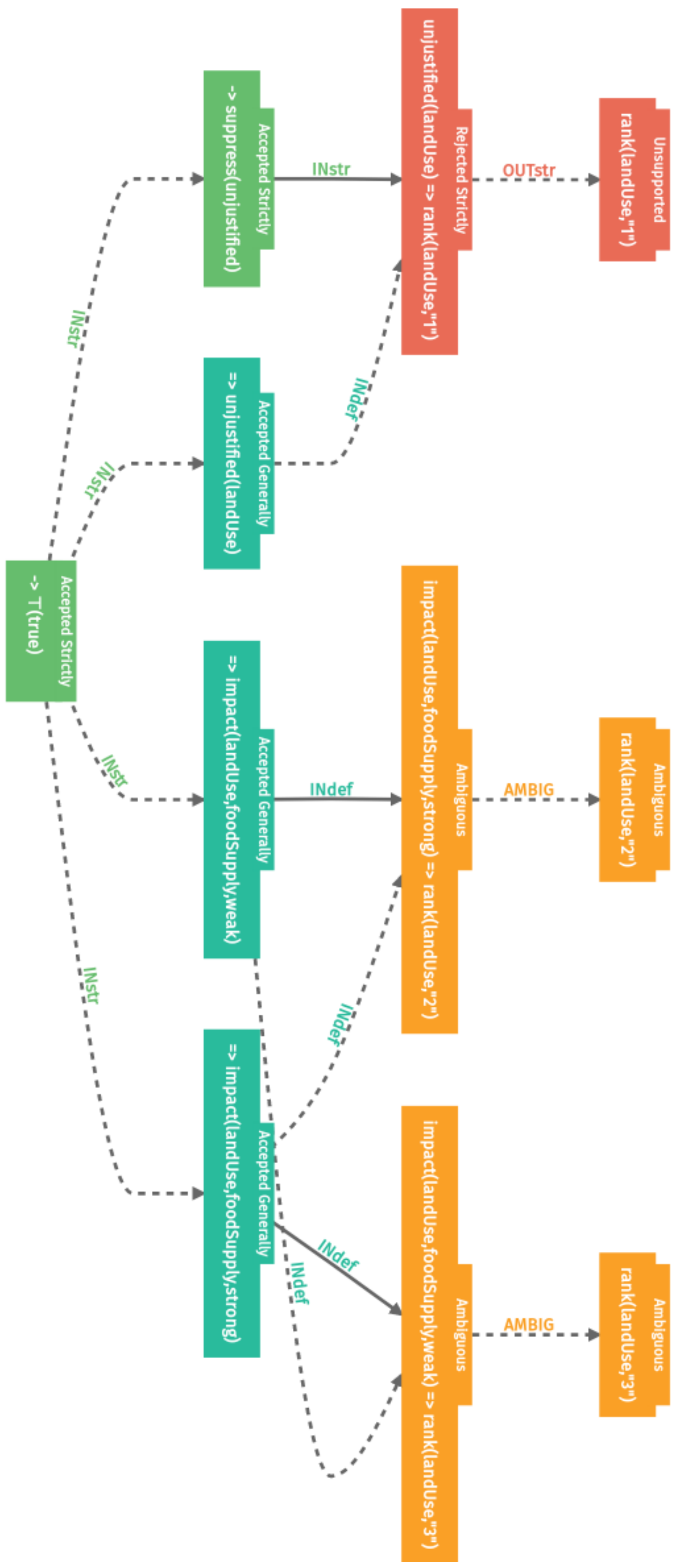


Figure 7. *Statement Graph for the landUse ranking and justifications.*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

*Without considering justifications, the aggregated ranking using for example the score voting method [Baujard et al., 2018] where each rank is given a score (rank 1 is 2 points, rank 2 is 1 point, and rank 3 is 0 point) would be "**globalWarming ~ landUse ~ waterConsumption**" where each alternative has a score of 3. This means that all alternatives are equivalent. However, by applying our rank sliding approach we get that:*

- *Global Warming alternative: (cf. Figure 5) both Alice's and Bob's rank for the globalWarming alternative have accepted justification, therefore they are not changed. However, Carol did not justify her ranking and thus her justification is rejected. Her rank of "3" for globalWarming should be changed to the closest rank with accepted justification, which is rank 2.*
- *Water Consumption alternative: (cf. Figure 6) Bob's rank for water consumption has an accepted justification therefore it is not changed. Alice's rank for water consumption has an ambiguous justification while Carol's rank has a rejected justification, thus they are both changed to the closest rank with accepted justification which is rank 3.*
- *Land Use alternative: (cf. Figure 7) both Alice's and Bob's rank for land use have an ambiguous justification, and since there is no rank with an accepted justification, they remain unchanged. Carol's rank has a rejected justification therefore it is changed to the closest admissible rank which is 2.*

The final rankings are as follows :

- *Alice's updated ranking: globalWarming is ranked 2, landUse and waterConsumption are ranked 3, we can represent this ranking into an order of alternative as follows : **globalWarming > landUse ~ waterConsumption**.*
- *Bob's updated ranking: globalWarming is ranked 1, landUse is ranked 2, and waterConsumption is ranked 3 i.e. **globalWarming > landUse > waterConsumption**.*
- *Carol's updated ranking: globalWarming and landUse are ranked 2, and waterConsumption is ranked 3 i.e. **globalWarming ~ landUse > waterConsumption**.*

*Using score voting method, we can find that globalWarming has a score of 4, landUse has a score of 2, and waterConsumption has a score of 0, therefore the final aggregated ranking is: **globalWarming > landUse > waterConsumption**.*

4.3. Implementation and tools

This section presents the general architecture and implementation of the decision-making platform. In the previous section we described the theoretical approach and formal pipeline behind our proposal and established three main objectives: (1) automatic reasoning about justifications, (2) establishing the status of justifications (accepted, rejected, or ambiguous), and (3) updating the rankings to produce justified rankings. While our general aim is to obtain justified collective decision making on alternatives, each one of these objectives can be seen as an end by itself. That is why we chose to adopt a flexible modular architecture of three modules: (1) Knowledge input, display and agent-collaboration module, (2) Reasoning module, and (3) Justified rankings module. Each module is a standalone tool that can act independently or be combined as a global platform to achieve our general aim. In the following we describe the general roles of each module and in the next sections we detail their implementation and added value.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

Knowledge input, display, and collaboration module. The role of this module is to collect the knowledge of the agents, display the Statement Graph, and allow these agents to collaborate in real time by updating their knowledge base. It is the *"entry point"* of our decision making platform, therefore it interacts with the agents (or a data engineer representing the agents) and exports the knowledge bases of each agent as a JSON file while also displaying the resulting Statement Graph produced by the reasoning module. **JSON** (JavaScript Object Notation) is a light weight data-interchange format that is widely used in the web.

Reasoning module. The role of this module is to build the support and attack links of the Statement Graph and give the status of the justifications (accepted, rejected, or ambiguous) depending on the chosen semantics. It is the *"brain"* of our decision making platform and takes as input the JSON file of the agents' knowledge bases and outputs the Statement Graph with or without the status of each reasoning step as a JSON object.

Justified rankings module. The role of this module is to update the rankings of each agents depending on the status of their justification. It represents the "decision making" part of our decision making platform. It takes as an input a JSON file describing the Statement Graph with the status of each reasoning step and exports an updated justified ranking for each agent.

4.3.1. Knowledge input, display, and collaboration module

This module has been implemented as a stand-alone tool called **DAMN** (Defeasible Reasoning With Statement Graph) available at the following link https://hamhec.github.io/damn (source code available at https://github.com/hamhec/damn). It is built using the client-server architecture where the **front-end** describes the interface that the user will interact with and the **back-end** describes the different servers that handle data storage, collaboration with other agents, and interaction with other modules (as described in Figure 12).

Module's workflow and functionalities. In this tool, every decision problem is called a "project", it can have multiple agents, each agent has his own knowledge base. A key difference must be made between an *"agent"* and a *"user"*. An agent is an entity that takes part in a decision making problem, it has its own knowledge base and can represent a person or a group of persons. A user is a person that interacts with the tool, it can handle different agents or represent one single agent. The typical workflow of this module (described in the activity diagram of Figure 8) is: (1) A user logs in or creates his account on the platform; (2) he can then create or open a previously created project; (3) The user can add or remove agents to the project along with their knowledge bases that can be imported from a DLGP file (a format used to describe rules and facts, more details are provided in the input format section); (4) He can invite other users to collaborate, insert, and modify the knowledge of one or multiple agents in real time. (5) The user can build the Statement Graph from some or all the knowledge bases and check the status of a justification by inputting a query and choosing a semantics; (6) He can then interact with the resulting displayed graph.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

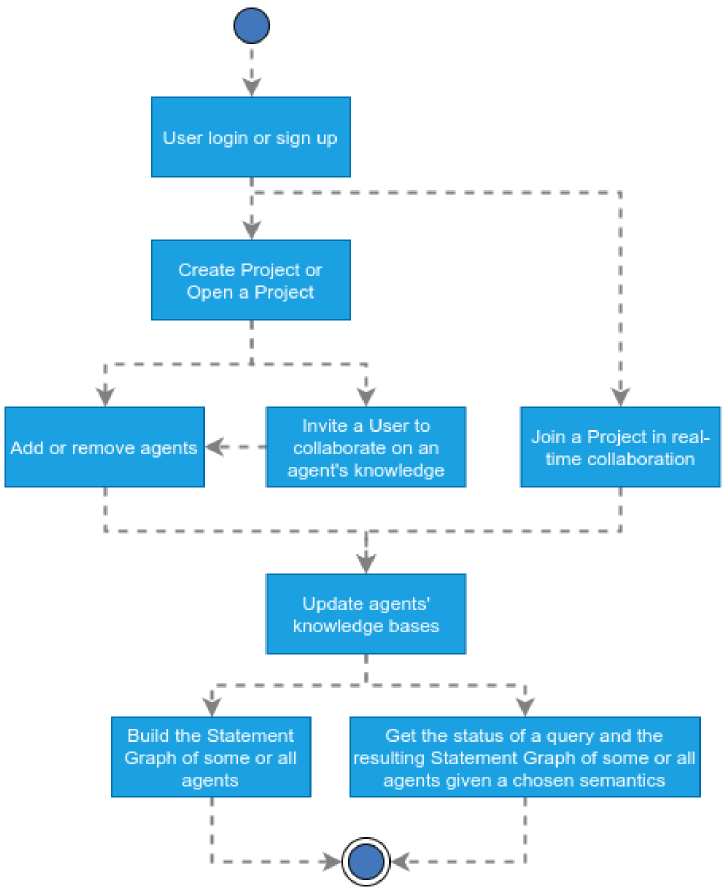

User login or sign up
Create Project or Open a Project
Add or remove agents
Invite a User to collaborate on an agent's knowledge
Join a Project in real-time collaboration
Update agents' knowledge bases
Build the Statement Graph of some or all agents
Get the status of a query and the resulting Statement Graph of some or all agents given a chosen semantics

Figure 8. A typical workflow of the DAMN tool.

This tool provides a data engineer with the following features:

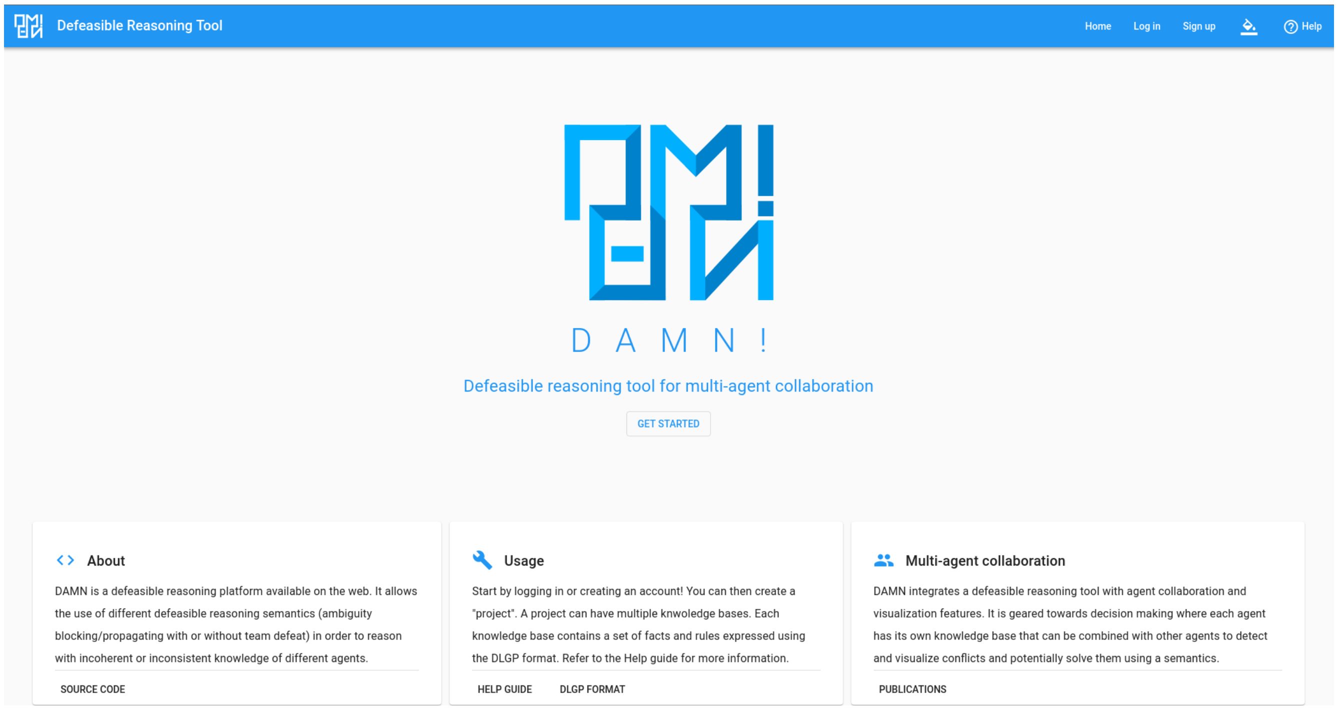
- Multi-agent: different knowledge bases for each agent with a common knowledge base describing the shared knowledge between all agents.
- Importing a project from a JSON file or importing the knowledge of each agent using a DLGP file.
- Real-time user collaboration with access control on what agents' knowledge can be updated by a certain user or users.
- Building the Statement Graph using some or all the knowledge bases of the agents, this can be used for example to check the support and attack links between a subset of agents.
- Obtaining the status (accepted, rejected, or ambiguous) of some or all reasoning steps by choosing a query and a semantics (ambiguity blocking or propagating with or without team defeat).
- Visualization of the Statement Graph along with interaction with the nodes (reasoning steps).
- Indication of origin for a reasoning step: each node indicates what agents helped in its creation (i.e. the set of agents that provided a rule or a fact that lead to this rule application). This indication of origin is useful to quickly understand which agents support a conclusion.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

- Exporting the agents’ knowledge bases in a JSON format.

Module’s Graphical User Interface. The UI is built from three main layout web pages: “**Home**”, “**Dashboard**”, and “**Project**”. In the “Home” layout the user can login/signup and check the different guides and help for the input format and how to use the tool. In the “Dashboard” layout, the logged-in user can create/delete/update his own projects or join collaboration projects with other users. In the “Project” layout the user can add agents, invite other users, input the knowledge, build or query the knowledge of some or all the agents, and interact with the resulting Statement Graph as shown in the following Example 6.

Example 6. *Consider the decision problem of Example 5. In order to build the statement graphs and visualize the status of the justification for the rankings a user has to: (1) Create or log into his account through the “login” or “sign up” in the upper right of the tool bar (cf. Figure 9); (2) create or open a previously created project (cf. Figure 10 upper right of the tool bar); (3) Add Alice, Bob, and Carol as agents and input their knowledge directly or import a DLGP file (cf. Figure 11upper left tool bar); (4) Build or query some or all the agents’ knowledge base by selecting them and either clicking the build button or adding a query and clicking the query button on the left of the screen (cf. Figure 11); (5) he can then use the graph menu to zoom in/zoom out, center the view on the graph, or even collapse certain nodes by double clicking on them (cf. Figure 11 right vertical menu).*


Defeasible Reasoning Tool
Home
Log in
Sign up
Help
D A M N !
Defeasible reasoning tool for multi-agent collaboration
GET STARTED
About
Usage
Multi-agent collaboration
SOURCE CODE
HELP GUIDE
DLGP FORMAT
PUBLICATIONS

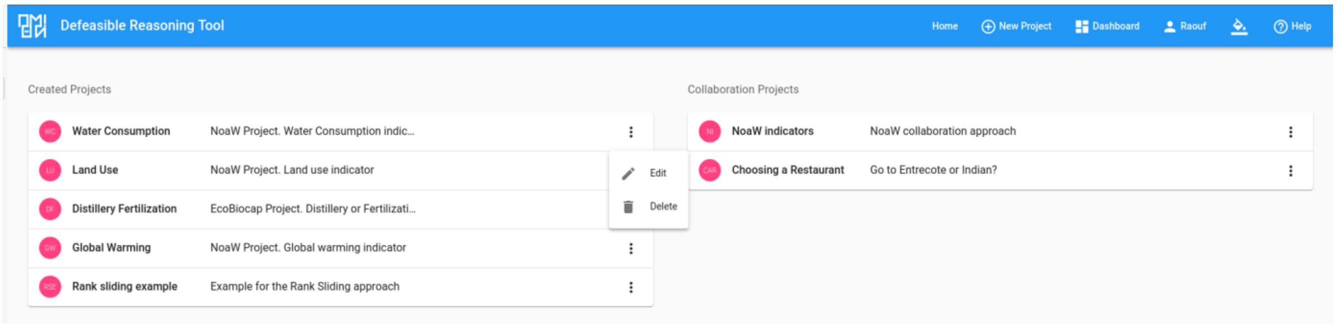
DAMN is a defeasible reasoning platform available on the web. It allows the use of different defeasible reasoning semantics (ambiguity blocking/propagating with or without team defeat) in order to reason with incoherent or inconsistent knowledge of different agents.

Start by logging in or creating an account! You can then create a "project". A project can have multiple knwoledge bases. Each knowledge base contains a set of facts and rules expressed using the DLGP format. Refer to the Help guide for more information.

DAMN integrates a defeasible reasoning tool with agent collaboration and visualization features. It is geared towards decision making where each agent has its own knowledge base that can be combined with other agents to detect and visualize conflicts and potentially solve them using a semantics.

Figure 9. Home webpage of the DAMN tool.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 688338



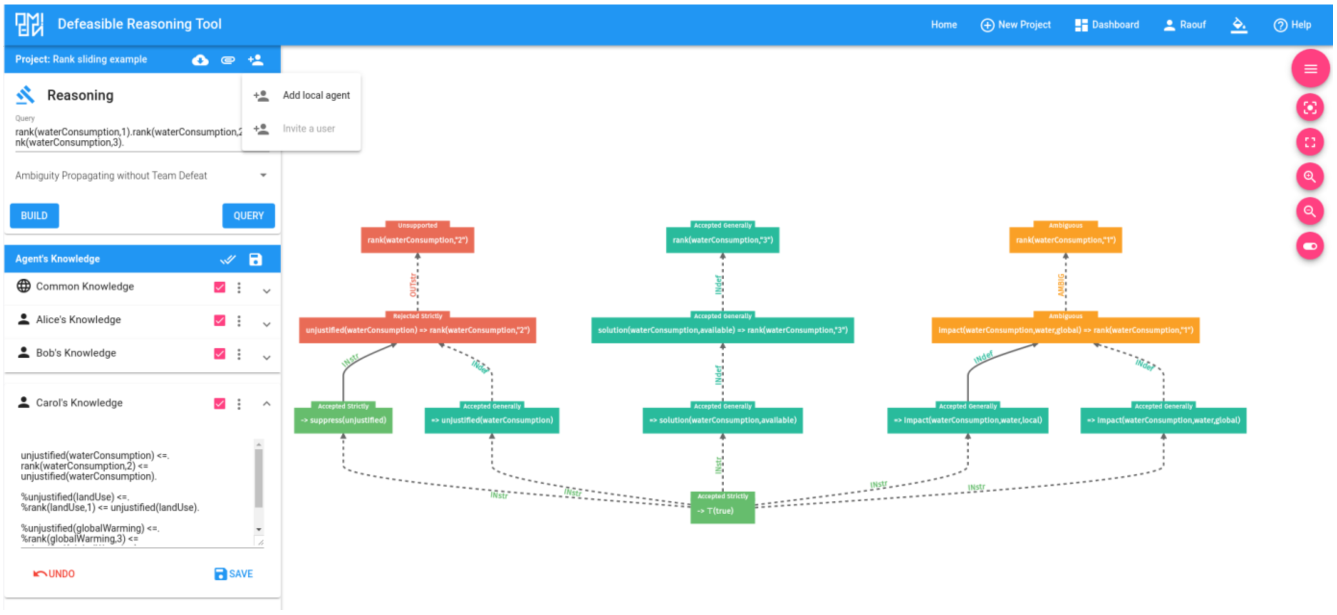
Created Projects

Water Consumption	NoaW Project. Water Consumption indic...
Land Use	NoaW Project. Land use indicator
Distillery Fertilization	EcoBiocap Project. Distillery or Fertilizati...
Global Warming	NoaW Project. Global warming indicator
Rank sliding example	Example for the Rank Sliding approach

Collaboration Projects

NoaW indicators	NoaW collaboration approach
Choosing a Restaurant	Go to Entrecote or Indian?

Figure 10. Dashboard webpage of the DAMN tool.


Defeasible Reasoning Tool
Home
New Project
Dashboard
Raouf
Help
Project: Rank sliding example
Reasoning
Add local agent
Invite a user
BUILD
QUERY
Agent's Knowledge
Common Knowledge
Alice's Knowledge
Bob's Knowledge
Carol's Knowledge
UNDO
SAVE
Unsupported
rank(waterConsumption,"2")
OUTstr
Rejected Strictly
unjustified(waterConsumption) => rank(waterConsumption,"2")
INstr
INdef
Accepted Strictly
-> suppress(unjustified)
Accepted Generally
=> unjustified(waterConsumption)
Accepted Generally
rank(waterConsumption,"3")
INdef
Accepted Generally
solution(waterConsumption,available) => rank(waterConsumption,"3")
INdef
Accepted Generally
=> solution(waterConsumption,available)
INstr
Ambiguous
rank(waterConsumption,"1")
AMBIG
Ambiguous
impact(waterConsumption,water,global) => rank(waterConsumption,"1")
INdef
INdef
Accepted Generally
=> impact(waterConsumption,water,local)
Accepted Generally
=> impact(waterConsumption,water,global)
INstr
INstr
INstr
INstr
Accepted Strictly
-> T(true)

Query
rank(waterConsumption,1).rank(waterConsumption,2
nk(waterConsumption,3).

Ambiguity Propagating without Team Defeat

```
unjustified(waterConsumption) <=.
rank(waterConsumption,2) <=
unjustified(waterConsumption).

%unjustified(landUse) <=.
%rank(landUse,1) <= unjustified(landUse).

%unjustified(globalWarming) <=.
%rank(globalWarming,3) <=
```

Figure 11. Project webpage of the DAMN tool.

Module's architecture. The DAMN tool architecture (cf. Figure 6) is composed of:

- **A client side application** (browser-based User Interface) for user interaction that displays the agents and their knowledge bases along with graph visualization features. It is built with Angular 7 which is a Typescript platform for building mobile and desktop web applications. The front-end is composed of various sub-modules including a custom built graph visualization module. The source code for the front-end angular application is available at https://github.com/ham-hec/damn.
- **A collaboration server** that handles the interaction between different users on the same project. It is built as a Websocket communication protocol using Spring Websocket. It ensures real-time broadcast of changes made by any user on a collaboration project.
- **A persistence server** that stores the data (built with a MongoDB server). MongoDB is a cross-platform document-oriented database program. Classified as a NoSQL database program that

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

stores data as JSON-like documents. This server stores the details about the different projects, as well as the information about the users.

- **A reasoning server** that contains the implementation of the reasoning module. It is used to obtain a JSON representation of the Statement Graph. More details are provided in the implementation section of this module.

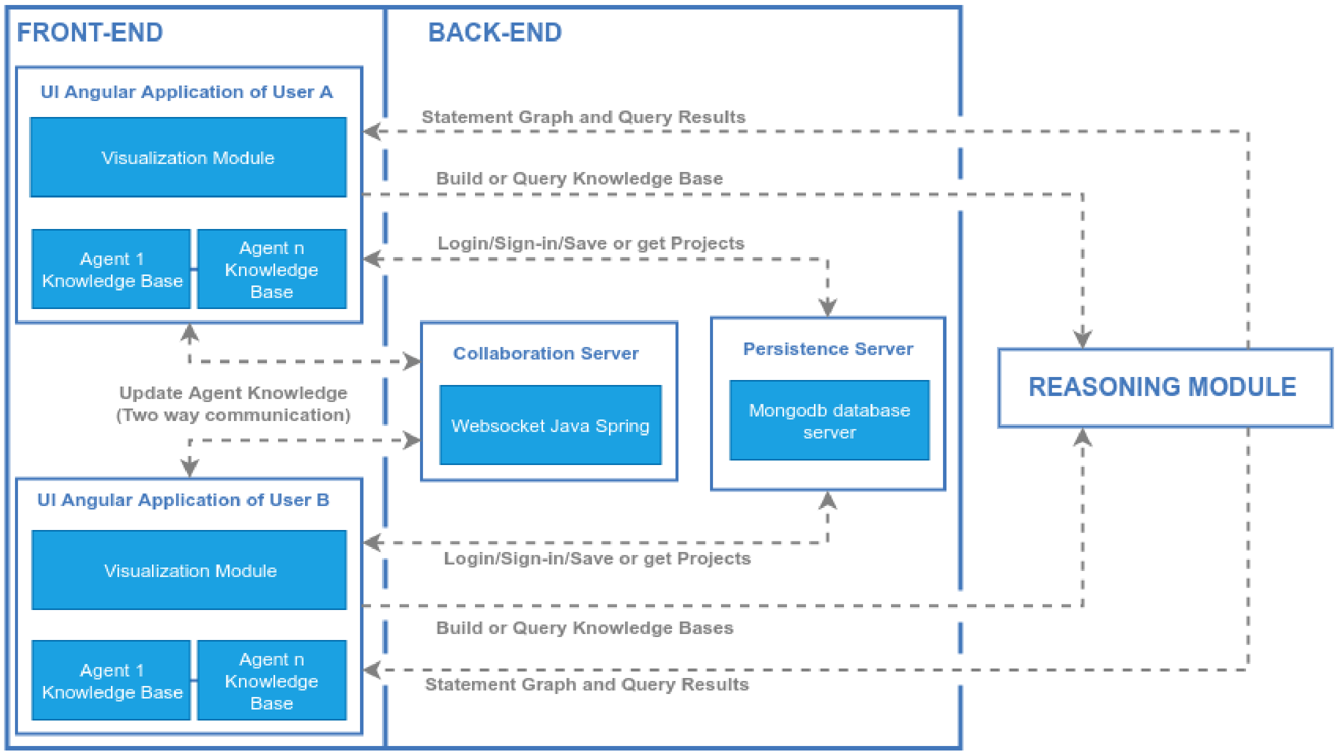

FRONT-END
BACK-END
UI Angular Application of User A
Visualization Module
Agent 1 Knowledge Base
Agent n Knowledge Base
Statement Graph and Query Results
Build or Query Knowledge Base
Login/Sign-in/Save or get Projects
Collaboration Server
Websocket Java Spring
Persistence Server
Mongodb database server
REASONING MODULE
Update Agent Knowledge (Two way communication)
UI Angular Application of User B
Visualization Module
Agent 1 Knowledge Base
Agent n Knowledge Base
Login/Sign-in/Save or get Projects
Build or Query Knowledge Bases
Statement Graph and Query Results

Figure 12. DAMN tool architecture.

Module's input and output format. There can be two ways to input a decision problem's knowledge bases: either through a JSON file that describes the whole project along with its knowledge bases, or through a DLGP file containing the facts and rules of one knowledge base. **DLGP format** [Baget et al., 2015] is a well-known format of expressing facts and rules in the Semantic Web and the Database domains:

- **Term:** a string representing either a variable (uppercase) e.g. X, or a constant (lowercase) e.g. bob. The formal grammar is <variable> = A..Z {A..Z | a..z | 0..9}*. <constant> = a..z | 0..9 {A..Z | a..z | 0..9}*. <term> = <variable> | <constant>.
- **Predicate:** a string starting with a lower case letter representing a relation between terms e.g. rank. The formal grammar is <predicate> = a..z {A..Z | a..z | 0..9}*.
- **Atom:** a string representing a predicate with terms between parenthesis e.g. rank(landUse,1). The formal grammar is: <atom> = <predicate>(<term> {, <term>}*).
- **Conjunction:** a list of comma separated atoms e.g. penguin(kowalski), age(2).The formal grammar is <conjunction> = <atom> {, <atom>}*.

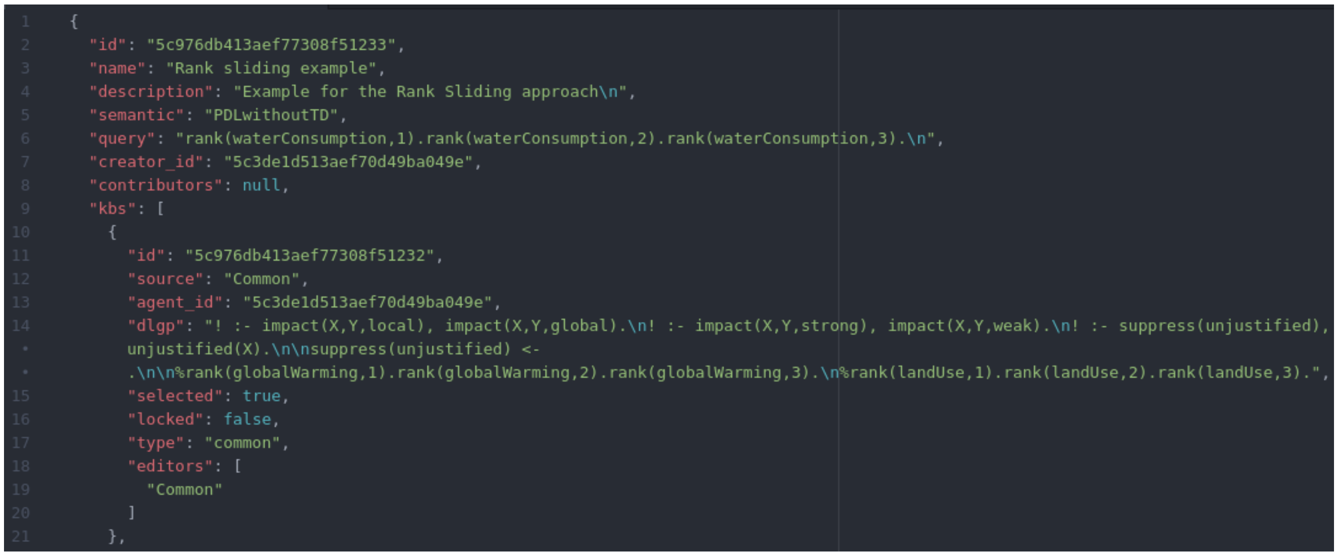
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

- **Rule label:** a string representing a name between brackets given to a rule e.g. [r1]. The formal grammar is: <label> = [A..Z | a..z | 0..9}*].
- **Rule:** a conjunction implied by conjunction, a rule can also have a label e.g. [r1] bird(X) <= penguin(X). The formal grammar is <strict_rule> = <label> <conjunction> <= <conjunction>.
- **Negative constraint:** a conjunction implying the logical contradiction e.g. *! :- fly(X), notFly(X)*. The formal grammar is <negative_constraint> = ! :- <conjunction>.
- **Preference between rules:** describes that a rule overrides another rule e.g. [r1] >> [r2]. The formal grammar is <preference> = <label> >> <label>.
- **Query:** a conjunction of atoms.

Example 7. *Consider the knowledge of Example 2. It can be expressed in DLGP format as follows:*
penguin(kowalski) <= . [r1] notFly(X), bird(X) <= penguin(X). [r2] fly(X) <= bird(X). r1 >> r2.
! :- fly(X), notFly(X).

A project can also be imported from a **JSON file**, this file should contain the following information: “name” (project name), “description”, “kbs” (a list of knowledge bases) every knowledge base contains the properties “source” (the name of the agent) and “dlgp” (the knowledge described in the dlgp format). The output format is a JSON file describing the project, it contains the same information as the JSON input format plus information about the users who contribute to the project, etc.

Example 8. *Part of the output JSON file of the decision problem described in Example 5 is as follows:*



```
{
  "id": "5c976db413aef77308f51233",
  "name": "Rank sliding example",
  "description": "Example for the Rank Sliding approach\n",
  "semantic": "PDLwithoutTD",
  "query": "rank(waterConsumption,1).rank(waterConsumption,2).rank(waterConsumption,3).\n",
  "creator_id": "5c3de1d513aef70d49ba049e",
  "contributors": null,
  "kbs": [
    {
      "id": "5c976db413aef77308f51232",
      "source": "Common",
      "agent_id": "5c3de1d513aef70d49ba049e",
      "dlgp": "! :- impact(X,Y,local), impact(X,Y,global).\n! :- impact(X,Y,strong), impact(X,Y,weak).\n! :- suppress(unjustified),
      unjustified(X).\n\nsuppress(unjustified) <-
      .\n\n%rank(globalWarming,1).rank(globalWarming,2).rank(globalWarming,3).\n%rank(landUse,1).rank(landUse,2).rank(landUse,3).",
      "selected": true,
      "locked": false,
      "type": "common",
      "editors": [
        "Common"
      ]
    },
```

Added value with regards to the state of the art. In this section, we discuss how our tool compares to the other tools available in the state of the art. Different tools for defeasible reasoning (conflict-tolerant reasoning) have been proposed in the literature, most notably, ASPIC+ [Prakken, 2010], DeLP [Garcia et al., 2004], DEFT [Hecham et al., 2017], Flora-2 [Wan et al., 2015], and SPINdle [Lam, 2012]. However, each tool allows for a different set of defeasible reasoning features as explained in [Hecham et al., 2018c], and none of them provides support for multi-agent collaboration or visualization as shown in the following Table 1.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 688338

Feature		ASPIC+	DeLP	DEFT	Flora-2	SPINdle		DAMN
Ambiguity	Propagating	✓	✓	✓	-	✓		✓
	Blocking	-	-	-	✓	✓		✓
Team Defeat	with	-	✓	✓	✓	✓		✓
	without	✓	-	-	-	-		✓
Existential Rules		-	-	✓	-	-		✓
Web-based		✓	✓	-	-	-		✓
Agent collaboration		-	-	-	-	-		✓
Visualization		-	-	-	-	-		✓

Table 1. Tools features (based on results from [Hecham et al., 2018c]) (✓indicates the tool supports the feature)

4.3.2. Reasoning module

This module has been implemented as a stand-alone tool called **ELDR** (Existential Logic for Defeasible Reasoning) available at the following link https://github.com/hamhec/eldr). It is a JAVA API uploaded on a REST server. This means that it is accessed not through a graphical interface but rather through the ability to call a set of functions on the web and obtain the resulting Statement Graph as JSON file.

Module's workflow and functionalities. This tool is packaged as an API (Application Programming Interface) and is available as a Maven module in the official public Maven repository (https://search.maven.org/artifact/fr.lirmm.graphik/graal-elder/1.0.17/jar). This means that the user can directly import this module and call a set of functions that offer the following functionalities:

- Parsing knowledge (set of rules and facts) from DLGP format to Java Objects that can later be used with the GRAAL reasoning tool (see next page).
- Building a Statement Graph (reasoning steps with support and attack links) from a JSON file describing a set of knowledge bases.
- Applying a semantics to a Statement Graph in order to obtain the status of a query or a set of queries.

Module architecture. The ELDR tool architecture (cf. Figure 13) is composed of:

- **Java Spring server** that exposes the API to the web and allows access to the functionalities via HTTP requests. Java Spring is an application framework for the Java platform, it provides the core features for constructing a web server that allows online use of any java application.
- **The API** which is a set of functions that can be called to build or query a Statement Graph. It takes as input a JSON description of the knoweldge bases and outputs a JSON file describing the Statement Graph.
- **A defeasible knowledge parser** that parses the knowledge base expressed using the DLGP format. This sub-module has been published as a stand alone Maven module for re-usability in

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

other projects (available at https://search.maven.org/artifact/fr.lirmm.graphik/graal-defeasible-core/0.0.6/jar). It's source code is available at (https://github.com/hamhec/graal-defeasible-core). It takes as input a DLGP string or file and outputs Java objects representing the rules and facts.

- **A Statement Graph builder** component that calls the parser to parse the rules and facts of the agents' knowledge bases expressed in DLGP format. The parsed objects are passed to the GRAAL reasoner to generate the reasoning steps (rule applications), then it builds the support and attack links between the statements. It takes as input a set of Java objects representing the rules, facts and the query and outputs a Java object representation of the Statement Graph.
- **A semantics** component that gives the reasoning steps a status (accepted, rejected, or ambiguous) depending on the chosen semantics (ambiguity blocking or propagating with or without team defeat). It takes as input a Java object representation of a Statement Graph along with the chosen semantics and outputs a JSON representation of the Statement Graph along with the statuses of its statements.
- **The GRAAL reasoner** (http://graphik-team.github.io/graal) which is a Java toolkit dedicated to querying knowledge bases within the framework of existential rules. It is used to generate the knowledge by applying the rules over the set of facts.

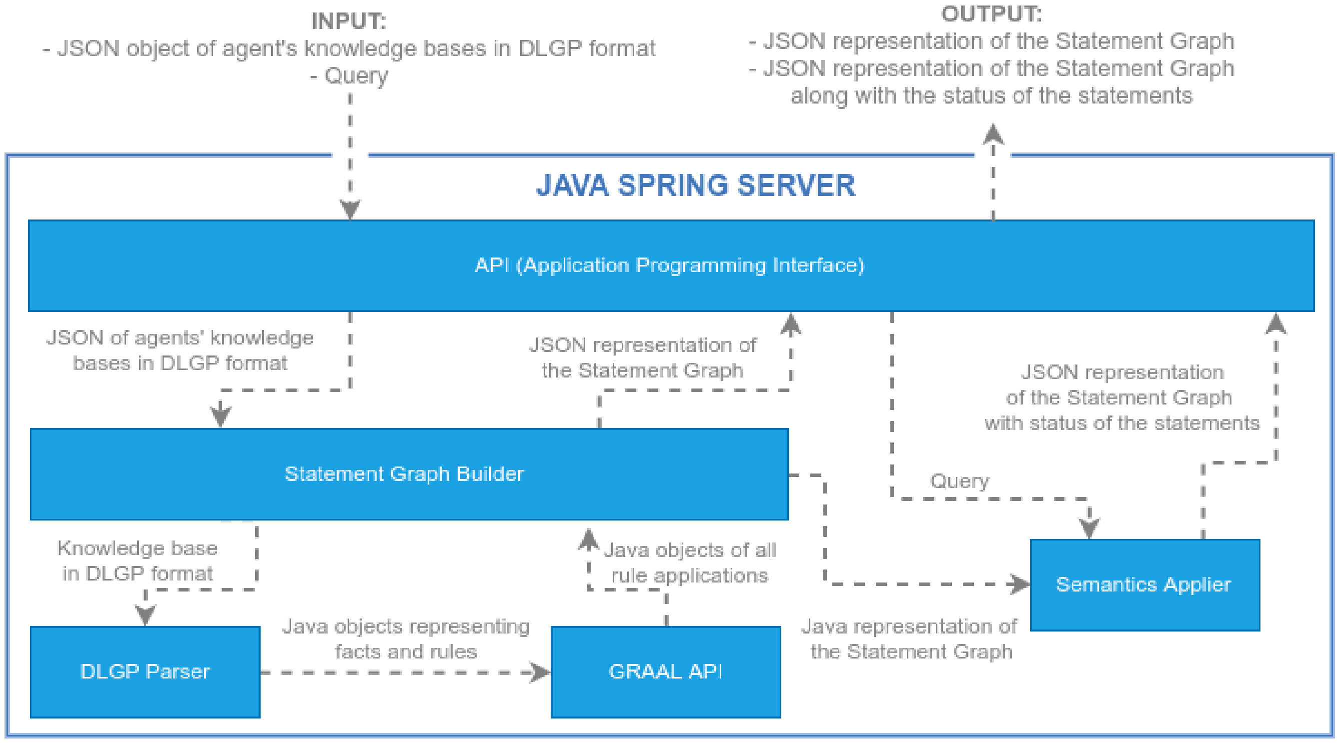

INPUT:
- JSON object of agent's knowledge bases in DLGP format
- Query
OUTPUT:
- JSON representation of the Statement Graph
- JSON representation of the Statement Graph along with the status of the statements
JAVA SPRING SERVER
API (Application Programming Interface)
JSON of agents' knowledge bases in DLGP format
JSON representation of the Statement Graph
JSON representation of the Statement Graph with status of the statements
Query
Statement Graph Builder
Knowledge base in DLGP format
Java objects of all rule applications
Semantics Applier
Java objects representing facts and rules
Java representation of the Statement Graph
DLGP Parser
GRAAL API

Figure 13. The ELDR tool architecture.

Module input and output format. The input of this module is a JSON object that describes a set of knowledge bases with optionally a set of queries. The JSON object should contain the following properties: a "query" property which is a dot-separated string of conjunction of atoms, a "kbs" attribute which

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

is a list of knowledge bases where each knowledge base contains the properties “source” (the name of the agent) and “dlgp” (the knowledge described in the dlgp format). The output format is a JSON object describing the resulting Statement Graph, it contains the property “statements” which is a list of statements and the “edges” property which is a list of links between statements. Each statement has an “id” property describing its unique identifier, a “title” property containing the rule application, and an “origin” property containing the list of agents that contributed to generate this rule application (the indication of origin is useful to quickly understand which agents support a conclusion). Each edge has a “source” and a “target” property containing the identifiers of the statements this edge links

Added value with regards to the state of the art. The performance of our tool has been tested in a benchmark [Hecham et al., 2018c]. It has been shown (among other things) that it offers one of the best reasoning time compared to other tools.

4.3.3. Justified ranking module

This module is a direct implementation of our rank sliding approach. The implementation of the collaboration approach has been included in the DAMN tool.

Module’s workflow and functionalities. This module is packaged as an API composed of two parts: a function that exposes the implementation of our rank sliding approach, and a set of abstract functions that can be used as a bases for future implementation of different rank changing approaches. Thus, this module serves two purposes:

- Provide a base implementation of any rank changing approach so that new approaches can be directly implemented
- Provide an implementation of our rank sliding approach.

Module architecture. This module exposes a single JAVA class with a function that takes as input the ranks of each agents and the status of the justification for these ranks, and outputs new ranks for each agent.

Module input and output format. The input of this module is a JSON object that represents the Statement Graph with the status of each justification and the knowledge bases of each agent. The output is a JSON object that gives for each agent the new ranking of the alternatives.

4.4. An Example on real Data: NoAW impact categories

In order to demonstrate the usability to our proposed approach and set of tools, we apply them on the data extracted in the context of the NoAW project. More precisely, a survey was conducted on LCA (Life Cycle Analysis) impact categories where NoAW stakeholders expressed their preferences and justifications about the importance (represented as ranks) of these categories. We are interested in assessing the resulting aggregating ranking with or without taking into account the potentially conflicting justifications.

The survey was conducted on 31 participants from the European projects NoAW and Agrocycle consisting of both public and private stakeholders (around 64% public researchers, 26% private companies, and 10% public agencies). They were asked to give a ranking of importance of 18 different impact categories along with their justification for this ranking. In this section we describe part of the results obtain by applying our proposed approaches and tools on this use-case.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 688338

4.4.1. Representing the justification extracted from the survey

The survey asked the different agents to provide their rankings on the impact categories and justify them. These justifications have been expressed as paragraphs or sentences. In order to fully exploit the powerful inference mechanism of logic showcased in the previous sections, we need to represent these justifications in a logical language. Knowledge representation is a domain by itself as it encapsulates a difficult task since many of the justifications rely on implicit knowledge that needs to be rendered explicit. Furthermore, the English language in which the survey was conducted is much more expressive than the logical language we use. That is why a set of predefined vocabulary (also called ontology) needs to be defined in order to express the justifications in a unified manner. This fixed vocabulary offers two main advantages:

1. A unified representation of all justifications, this makes the automatic detection of agreements and disagreements easier.
2. The ability to automatically translate the reasoning steps expressed in the logical language to a sentence expressed in human language (English, French, etc.). For example the atom "impact(waterConsumption, freshwater, negative, strong, local)" can be translated to: "Water consumption has a local strong negative impact on freshwater", this can be expressed in an abstract way as "impact(X,Y,Z,U,V)" is translated to "X has a V,U,Z impact on Y".

Given the different justifications with their explicit and implicit knowledge expressed in the survey, we have established the following vocabulary (set of predicates and atoms):

- "**impact(Category,Something,Type,Strength,Level)**" to express the *strength* (strong, medium, weak) and *type* (positive, neutral, negative) of the impact that an impact *category* has on *something* (e.g. freshwater, environment, etc.) at a specific *level* (local, global, etc.). It is translated to "*Category* has a *Level*, *Strength*, *Type* impact on *Something*". For example "impact(landUse, foodSupply, negative, weak, global)" which translates to "landUse has a global weak negative impact on foodSupply".
- "**accordingTo(Source,Something,Adjective)**" to express that a Source (e.g. media, own knowledge, etc.) has qualified Something (e.g. landUse) by an Adjective (e.g. important, etc.). It is translated to "According to Source, Something is Adjective". For example "accordingTo(media, landUse, veryImportant)" which translates to "According to media, landUse is veryImportant".
- "**solution(Problem,Type,Adjective)**" to express that there is a Adjective (e.g. available, easy, etc.) solution to the Problem (e.g. deforestation, etc.) with a specific Type (e.g. political, technological, etc.). It is translated to "A *Type* solution to the *Problem* problem is *Adjective*". For example "solution(freshwaterEcotoxicity,technological,available)" which translates to "A technological solution to the freshwaterEcotoxicity problem is available)".
- "**action(Action,Adjective)**" to give an action an adjective. It is translated to "Action is Adjective". For example, "action(increasingLandConsumption,unsustainable)" which translates to "*increasingLandConsumption* is *unsustainable*".
- "**essentialFor(Something1,Something2)**" to express the strong link between certain concepts. It is translated to "Something1 is essential for Something2". For example, "essentialFor(land,foodProduction)" which translates to "land is essential for foodProduction".

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

- "**relatedTo(Category,Something)**" to express the different concepts that are related to an impact category. It is translated to "Category is related to Something". For example, "relatedTo(landUse,deforestation)" which translates to "*landUse* is related to *deforestation*".
- "**cause(Something1,Something2)**" to express that Something1 causes Something2. It is translated to "Something1 causes Something2". For example "cause(populationGrowth,increasedDemandForWater)" which translates to "populationGrowth causes increasedDemandForWater".
- "**status(Something,Adjective,Level)**" to express the state of Something with an Adjective and a Level (global, local, etc.). It is translated to "The status of Something is Adjective on a Level level". For example, "status(freshwater,rare,global)" which translates to "the status of freshwater is rare on a global level".
- Other case-specific predicates have been added depending on the needs, such as "sameAs", "moreImportantThan", "lessImportantThan", etc.

Given this fixed vocabulary, we can define common sense contradictions such that a category cannot have the same type of impact on the same thing with a different strength. For example, "impact(landUse, foodSupply, negative, weak, global)" is in contradiction with "impact(landUse, foodSupply, negative, strong, global)", the same can be said for the level of impact (local vs global). Please refer to the common knowledge base of the JSON output for a list of common sense contradictions.

Justifications provided by the agents need to be manually expressed using the previously mentioned fixed set of vocabulary as shown in the following Example 9.

Example 9. *Consider the following justifications given by the different agents in the NoAW projects.*

- *"Water will be more precious due to population growth" as a justification for giving "Water Consumption" the rank of 3. This is translated to "status(water,rare,unspecified) ∧ cause(populationGrowth,increasedDemandForWater) ∧ impact(waterConsumption,freshwater,negative,unspecified,unspecified) ⇒rank(waterConsumption,3)"*

- *"Deforestation has to be stopped!" as a justification for giving "Land Use" the rank of 1. This is translated to "action(stoppingDeforestation,necessary) ∧ relatedTo(landUse,deforestation) ⇒ rank(landUse,1)."*

The interested reader is referred to the JSON output file and the survey data file for more details.

Expressing human knowledge in a logical language is a well-known difficult task of knowledge representation, especially when implicit knowledge is involved. It requires a constant input from the agents and from the data engineer that represents their knowledge. That is why the current representation of the justifications extracted from the survey data can be improved upon if the involved agents could have discussed the final representation.

4.4.1. The results of applying the proposed approach

We have translated the justifications given by the different agents for 5 impact categories (global warming, water consumption, land use, freshwater eutrophication, and human carcinogenic toxicity). These categories were selected because they contained the most differences between agents' rankings. Future work is needed to translated the justifications for all the 18 impact categories.

Given that the agents were not available to perform a discussion phase, the only option we have is to use the rank sliding approach on the rankings with the current justifications. After representing the justifications using the ontology previously defined, we build and query the Statement Graphs of each alternative to obtain the status of the justifications. The results are as follows :

- Directed aggregation of rankings (without consideration of justifications): we apply the scoring voting method where each rank is given a score (rank 1 has a score of 18, rank 2 has a score of 17, etc, rank 18 has a score of 1). The final score of each alternative is simply the sum of the agents' score. The aggregated ranking is computed by simply ordering the impact categories from highest score to lowest score:

Impact Category	Total Score
Global Warming	477
Water Consumption	407
Land Use	409
Freshwater Eutrophication	372
Human Carcinogenic Toxicity	463

  Therefore we can conclude that the final ranking is:

  Global Warming > Human Carcinogenic Toxicity > Land Use > Water Consumption > Freshwater Eutrophication.

- With consideration of justifications: by applying our rank sliding approach we observe that 29 individual rank changes were performed, 25 of which were due to lack of justification (rejected justification) and 4 were due to conflicting justifications (ambiguous justification) (cf. Table 2, for more details, the interested reader is referred to the rank sliding data available at https://data.inra.fr/dataset.xhtml?persistentId=doi:10.15454/XVL4BA). Ranks with rejected justifications were dismissed which means they were changed to rank 19 and given 0 points in the score voting method. Ranks with ambiguous justification were changed to the closest rank with an accepted justification. For instance, two agents gave "Water Consumption" the rank 5 with ambiguous justifications, so their ranks were changed to 4 because it was the closest rank with an accepted justification. It is worth noting that in some cases, agents had ambiguous justifications for their ranks, however, their ranks were not changed because other agents were able to provide accepted justifications for the same ranks. Future work is recommended to take into account these special cases. The final results after rank changes are :

Impact Category	Total Updated Score
Global Warming	435
Water Consumption	345
Land Use	328
Freshwater Eutrophication	274
Human Carcinogenic Toxicity	358

Therefore, the final ranking after considering the justification is:

Global Warming > Human Carcinogenic Toxicity > Water Consumption > Land Use > Freshwater Eutrophication.

We can observe that the impact category « Land Use » changed position with « Water Consumption » mainly due to the lack of valid justifications provided by the agents. This result is promising because it shows that the rank sliding approach pushes the agents to not only justify their ranking but also discuss other agent rankings. This would allow them to keep their ranking while challenging other agents' justifications. Non justified rankings are then dismissed as personal preferences and more grounded rankings are favored.

	Global Warming		Water Consumption		Land Use		Freshwater Eutrophication		Human Carcinogenic Toxicity	
	Without justification	With justification	Without justification	With justification	Without justification	With justification	Without justification	With justification	Without justification	With justification
Agent 1	1	1	5	4	11	11	3	3	1	1
Agent 2	1	1	4	4	10	10	5	5	1	1
Agent 3	1	1	6	6	1	1	8	8	3	3
Agent 4	4	4	6	6	6	6	5	5	1	1
Agent 5	3	3	12	12	16	15	5	5	1	1
Agent 6	3	3	7	7	6	6	5	5	1	1
Agent 7	1	1	2	2	2	2	4	19	3	19
Agent 8	1	1	2	2	8	8	7	7	3	3
Agent 9	18	19	1	19	1	19	1	19	1	19
Agent 10	1	1	1	1	2	19	3	19	2	19
Agent 11	1	1	2	2	3	3	14	14	5	19
Agent 12	1	1	3	3	5	5	5	5	3	3
Agent 13	3	3	7	7	7	7	6	6	1	1
Agent 14	10	19	3	19	3	19	16	19	10	19
Agent 15	1	1	18	18	15	15	18	19	1	1
Agent 16	1	1	15	16	2	2	12	12	4	4
Agent 17	1	1	2	2	4	4	7	7	6	6
Agent 18	11	11	2	2	1	1	6	6	3	3
Agent 19	2	2	3	3	1	1	9	9	15	13
Agent 20	1	1	18	18	5	5	9	9	3	3
Agent 21	11	11	1	1	2	2	4	4	8	8
Agent 22	6	6	3	3	2	2	12	12	13	13
Agent 23	1	1	1	1	1	1	1	1	1	1
Agent 24	4	19	5	19	3	19	11	19	2	19
Agent 25	2	19	4	19	7	19	1	19	8	19
Agent 26	3	3	1	1	6	6	11	19	4	4
Agent 27	8	8	16	16	11	11	3	3	14	19
Agent 28	2	2	5	4	1	1	3	3	4	4
Agent 29	4	4	10	10	18	18	12	12	1	1
Agent 30	2	2	4	4	4	4	3	3	1	1
Agent 31	3	3	13	13	16	19	8	19	2	2
Score	477	435	407	345	409	328	372	274	463	358

Table 2. Overview of the ranks given by each agent with the potential changes (orange cells mean that the status of the justification is ambiguous, red cells mean that the status of the justification is rejected).

5. Conclusions

Summing up, in this deliverable we have proposed an approach and a set of software tools for collective decision-making by evaluating the justifications of the rankings on different alternatives. The workflow of the proposal is to first ask the agents to provide a justification for each of their ranks, then to automatically detect agreements and disagreements between the reasoning steps provided by these agents. Afterwards, they can discuss the points of conflicts and potentially change their rankings or justifications. Next, the argumentation technique known as Statement Graphs is applied to establish accepted, ambiguous, and rejected justifications which would be later used to filter and update the initial preferences using a rank sliding approach. Then a voting method from social choice can be used to aggregate these updated ranks.

With these objectives in mind we proposed a theoretical approach along with its supporting tools:

- Automatic reasoning about justifications: we have defined Statement Graphs which are graph that represent the agreements and disagreements between “reasoning steps”. We provided an implementation of this formalism in a tool called DAMN which offered the ability to visualize and interact with these graphs, along with the possibility to collaborate with other users.
- Establishing the status of justifications: we have implemented a tool called ELDR which provides a set of functions for building and querying Statement Graphs.
- Updating the ranks according to the justifications: we have proposed an approach called rank sliding that defines the general desirable properties of any approach that changes ranks. We also provide an implementation for such approach.

While the proposed approach and tools were designed to work in the general case, we have showcased the applicability of our method by applying it on the LCA (Life Cycle Analysis) impact categories use-case. More precisely, a survey on 31 participants from the European projects NoAW and Agrocycle was conducted on LCA impact categories where the stakeholders expressed their preferences and justifications about the importance of these categories. We assessed the resulting aggregated ranking with or without taking into account the potentially conflicting justifications. We showed that the final result changes and this indicates that applying our approach would yield various benefits such as incentivizing agents to provide justifications of their rankings and to discuss other agents’ justifications in order to avoid their vote being modified or dismissed.

For future work (not included in NoAW), we want to further extend our research in three main areas:

- On the theoretical side, we want to study different variants of rank sliding approaches along with other conflict-tolerant reasoning semantics such as a semantics that takes into consideration the number of supports and attacks for a justification. We also would like to study the possibility to generate automatically summaries of arguments that justify the final ranking.
- On the practical side, we want to extend the number of format that our tools can handle along with their usability, especially for the DAMN tool, by providing more options for manipulating and interacting with the graph along with the automatic translation from the logical language to a simplified human language.
- On the application side, we want to study the other impact categories and represent the knowledge associated to them, and to apply our approach to other use-cases in order to check its flexibility.

6. Partners involved and Dissemination

In order to improve the quality of the data collection our INRA team has collaborated with the following partners for the deployment of this deliverable. The partners involved in the work are Danmarks Tekniske Universitet (DTU - Technical University of Denmark – Denmark), Campden BRI Magyarorszag Nonprofit Korlatolt Felelossegu Tarsasag (CBHU – Hungary), and Université de Montpellier (UM – University of Montpellier - France).

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 688338

These partners helped in the design and implementation of the surveys (survey regarding viniculture and viticulture, and survey regarding LCA impact categories). In future work, we plan to continue our collaboration with DTU in order to assess the benefits of our approach in the LCA analysis.

Our proposed approach has been published in the following articles:

- Hecham, A., Bisquert, P., & Croitoru, M. (2018, July). « *On a Flexible Representation for Defeasible Reasoning Variants* ». In Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (pp. 1123-1131). International Foundation for Autonomous Agents and Multiagent Systems.

- Hecham, A., Bisquert, P., & Croitoru, M. (2018, September). « *Demonstrating a Benchmark for Defeasible Reasoning* ». In Proceedings of Computational Models of Argument - COMMA 2018 (pp. 461-462). Springer.

- Hecham, A., Croitoru, M., & Bisquert, P. (2018, September). *« A First Order Logic Benchmark for Defeasible Reasoning Tool Profiling »*. In International Joint Conference on Rules and Reasoning (pp. 81-97). Springer.

7. NoAW FAIR data management

Data produced in this deliverable have been stored in a dataset (https://data.inra.fr/dataset.xhtml?persistentId=doi:10.15454/XVL4BA) in conformity with NOAW FAIR data management policy.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

8. Bibliographic References

[Amgoud et al., 2009] Amgoud, L., Prade, H., (2009). 'Using arguments for making and explaining decisions'. *Artificial Intelligence,* vol. 173, no. 3-4, pp. 413-436

[Antonio et al, 2000] Antoniou, G., Billington, D., Governatori, G., Maher, M. J., & Rock, A. (2000, August). A family of defeasible reasoning logics and its implementation. In ECAI (Vol. 2000, pp. 459-463).

[Baader et al., 2005] Baader, F., Horrocks, I., and Sattler, U. (2005). De- scription logics as ontology languages for the semantic web. In Mechanizing Mathematical Reasoning, pages 228–248. Springer.

[Baget et al., 2015] Baget, J. F., Gutierrez, A., Leclere, M., Mugnier, M. L., Rocher, S., & Sipieter, C. (2015, August). Datalog+, RuleML and OWL 2: Formats and Translations for Existential Rules. In Challenge+ DC@ RuleML.

[Bauhard et al., 2018] Baujard, A., Gavrel, F., Igersheim, H., Laslier, J. F., & Lebon, I. (2018). How voters use grade scales in evaluative voting. European Journal of Political Economy, 55, 14-28.

[Bisquert et al., 2016] Bisquert, P., Croitoru, M., de Saint-Cyr, F. D., & Hecham, A. (2016, August). "Substantive irrationality in cognitive systems". In ECAI: European Conference on Artificial Intelligence (No. 285, pp. 1642-1643).

[Cali et al., 2010] Cali, A., Gottlob, G., Lukasiewicz, T., Marnette, B., and Pieris, A. (2010). Datalog+/-: A family of logical knowledge representation and query languages for new applications. In Logic in Computer Science (LICS), 2010 25th Annual IEEE Symposium on, pages 228–242. IEEE.

[Carnielli et al., 2001] Carnielli, W. A., & Marcos, J. (2001). Ex contradictione non sequitur quodlibet. Bulletin of Advanced Reasoning and Knowledge, 1, 89-109.

[Garcia et al., 2003] García, A. J., & Simari, G. R. (2003). Defeasible logic programming: An argumentative approach. Theory and practice of logic programming 4 (pp 95-138).

[Hecham et al., 2018a] Hecham, A., Bisquert, P., & Croitoru, M. (2018, July). « *On a Flexible Representation for Defeasible Reasoning Variants* ». In Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (pp. 1123-1131). International Foundation for Autonomous Agents and Multiagent Systems.

[Hecham et al., 2018b] Hecham, A., Bisquert, P., & Croitoru, M. (2018, September). « *Demonstrating a Benchmark for Defeasible Reasoning* ». In Proceedings of Computational Models of Argument - COMMA 2018 (pp. 461-462). Springer.

[Hecham et al., 2018c] Hecham, A., Croitoru, M., & Bisquert, P. (2018, September). *« A First Order Logic Benchmark for Defeasible Reasoning Tool Profiling »*. In International Joint Conference on Rules and Reasoning (pp. 81-97). Springer.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338

[Hecham et al., 2017] Hecham, A., Croitoru, M., & Bisquert, P. (2017). Argumentation-based defeasible reasoning for existential rules. In Proceedings of the international Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS, Vol. 3. (pp 1568-1569).

[Lam, 2012] Lam H. P. (2012). On the Derivability of Defeasible Logic. Thesis.

[Prakken, 2010] Prakken, H. (2010). An abstract framework for argumentation with structured arguments. Argument and Computation, 1(2), 93-124.

[Roy, 1991] Roy, B, 1991 'The outranking approach and the foundations of electre methods'. *Theory and Decision*, vol. 31, no. 1, pp. 49–73

[Von Winterfeldt et al., 1986] Von Winterfeldt, D, & Edwards, W, 1986, Decision analysis and behavorial research, Cambridge University Press, Cambridge

[Wan et al, 2015] Wan, H., Kifer, M., & Grosof, B. (2015). Defeasibility in answer set programs with defaults and argumentation rules. Semantic Web, 6(1), 81-98.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688338